FASTVLM2: EFFICIENT ON-DEVICE DENSE GROUNDING FOR GENERALIST VLMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Generalist vision-language models (VLMs) with dense grounding capabilities can perform a broad range of classical vision tasks through a single model and natural-language interface. However, efficient on-device inference with such models faces two complementary bottlenecks: costly high-resolution visual encoding and lengthy autoregressive decoding of numerous spatial locations. We introduce **FastVLM2**, a unified framework that addresses both. For *input-side* efficiency, we introduce a native-resolution strategy that adapts pretrained convolutional and hybrid convolution-transformer backbones (ConvNeXt and FastViTHD, respectively) through VLM instruction tuning, enabling arbitrary image resolutions and aspect ratios without additional vision pretraining, image tiling, or explicit layout tokens. On a MacBook with an M1 Max, FastVLM2-4B achieves VQA accuracy comparable to Qwen3-VL-4B with $4.6\times$ lower prefill latency. For *output-side* efficiency, we use quantized location tokens to compactly represent grounding outputs, reducing the output-token budget by over 60% relative to textual coordinate representations. To recover the localization accuracy lost to quantization, we introduce a simple Gaussian distance-aware soft cross-entropy objective that brings quantized grounding close to textual-coordinate performance while preserving this decoding advantage. At higher grounding accuracy, FastVLM2-1.8B achieves $7.3\times$ lower end-to-end latency than Rex-Omni. We further introduce a training-free, geometry-aware box confidence estimator that directly exploits the predictive distributions of quantized location tokens. We show that this scoring rule generalizes across three independently trained grounding models and consistently improves COCO mAP by more than 8 points over the score-free ranking protocol used in Rex-Omni. Together, FastVLM2 makes accurate, confidence-aware dense grounding practical for on-device generalist VLMs.

1 INTRODUCTION

The efficiency of a Vision-Language Model (VLM) hinges on the cost of processing input images through a vision encoder and generating structured output with a Large Language Model (LLM). On the input side, image resolution and aspect ratio naturally vary, yet most VLMs operate at fixed resolutions. Increasing resolution, for example through multi-crop processing, raises vision-encoding and LLM-prefill cost. Alternatively, VLMs can support images at their native resolution by modifying the vision encoder architecture and training. Native-resolution processing has primarily been explored with isotropic architectures such as ViTs (Dehghani et al., 2023), which maintain a constant spatial resolution across all layers, whereas low-latency hierarchical encoders with multi-stage spatial downsampling remain comparatively underexplored (Vasu et al., 2025).

We investigate "**Can a pretrained hierarchical encoder, originally trained at fixed resolution, acquire native-resolution capability solely during VLM fine-tuning?**" and answer affirmatively across both convolutional and convolution-transformer hierarchical encoders. We inject explicit two-dimensional coordinates into convolutional features using CoordConv (Liu et al., 2018); for convolution-transformer hybrids, we additionally use standard 2D RoPE (Heo et al., 2024) in the attention stages. These lightweight interventions are introduced only during VLM instruction tuning, without repeating large-scale vision pretraining. We demonstrate successful adaptation for both ConvNeXt and FastViTHD, showing that pretrained hierarchical backbones can be made resolution-flexible with minimal architectural changes.

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.

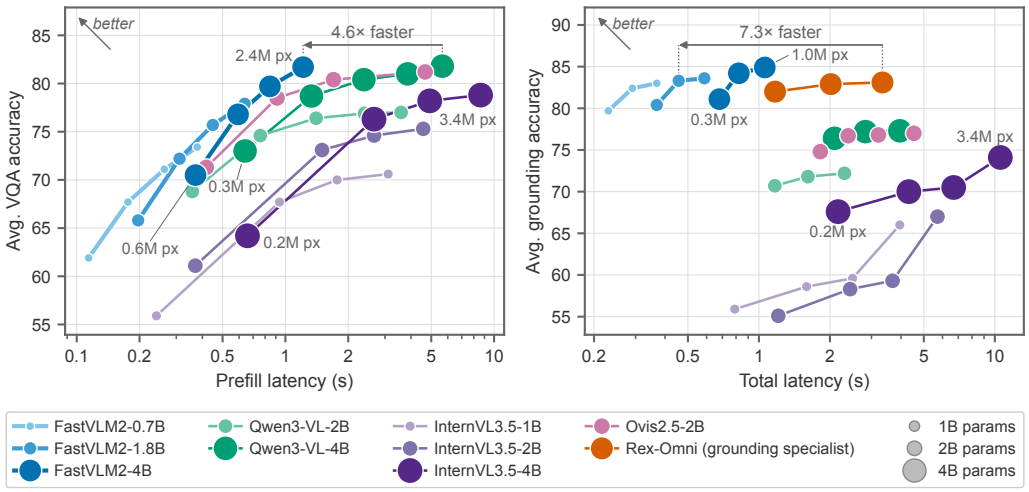


Figure 1: **FastVLM2 improves the accuracy-latency trade-off for VQA and grounding.** On a MacBook M1 Max, FastVLM2-4B has 4.6× lower prefill latency than Qwen3-VL-4B at comparable VQA accuracy (**left**), while FastVLM2-1.8B has 7.3× lower end-to-end latency than Rex-Omni at higher grounding accuracy (**right**). Models are evaluated at multiple operating points by varying the maximum number of input pixels, in millions (M px). Details are in Appx. G.

A second bottleneck lies in structured output generation for visual grounding. Generating bounding boxes as textual coordinates is natural but relatively token-intensive (Chen et al., 2023; Zhan et al., 2024). Quantized location tokens reduce decode cost with fewer tokens, but naive quantization introduces a measurable localization gap. We ask "**Can we retain the efficiency of quantized coordinates while matching the grounding accuracy of textual coordinate representations?**" We find that a simple Gaussian distance-aware soft cross-entropy objective recovers most of the grounding accuracy lost to quantization. This enables quantized coordinates to approach the accuracy of textual representations while retaining their inherent advantage of over 60% fewer output tokens. This is achieved without a task-specific detection head or inference-time architectural changes. Figure 1 summarizes the input and output accuracy-latency trade-offs.

Quantized location tokens offer the extra advantage of producing a distribution over spatial locations for each coordinate. This raises a second question: "**Can location distributions be converted into a meaningful box-level confidence score without training an auxiliary model or detection head?**" We show that they can. Rather than interpreting token probability or entropy directly in the quantized coordinate space, we map the coordinate distributions through the geometry of the predicted box and measure the probability mass assigned to geometrically consistent alternatives. The resulting confidence score is entirely training-free, requiring no additional parameters or forward passes, and remains consistent across quantized-location grounding models. Applying the same scoring rule to our model, Rex-Omni, and LocateAnything-3B consistently improves COCO mAP by ∼8.5 points over score-free ranking. This suggests that confidence estimation is a general capability of VLMs that represent boxes with discrete location tokens, rather than a model-specific mechanism.

In summary, our main contributions are:

- We introduce a lightweight approach for adapting pretrained hierarchical vision encoders to native-resolution inputs using only VLM fine-tuning, and demonstrate it on both convolutional (ConvNeXt) and convolution-transformer (FastViTHD) backbones, enabling them to operate across a continuum of accuracy-latency trade-offs.
- We close much of the localization gap between quantized and textual coordinate representations using Gaussian distance-aware soft cross-entropy, enabling quantized grounding to approach textual-coordinate performance while retaining over 60% fewer output tokens.
- We introduce a training-free, geometry-aware confidence estimator for boxes represented by discrete location tokens. It transfers without refitting across three grounding models, improving COCO mAP by ∼8.5 points over score-free ranking with no extra parameters or forward passes.

2 Related Work

Adaptive and native-resolution visual encoding. Most vision-language models initialize their vision encoders from CLIP-style contrastive pretraining (Radford et al., 2021; Vasu et al., 2024; Faghri et al., 2025; Tschannen et al., 2025), where images are typically processed at a fixed square resolution. A common alternative retains the fixed-resolution encoder and decomposes high-resolution images into multiple crops or tiles, as in SPHINX (Lin et al., 2024), LLaVA-NeXT (Liu et al., 2024b), LLaVA-OneVision (Li et al., 2025a), InternVL3.5 (Wang et al., 2025), and Molmo (Deitke et al., 2025; Clark et al., 2026). While these approaches preserve pretrained visual representations and expose finer image detail, processing multiple regions increases vision computation and the number of visual tokens presented to the language model. Some methods add explicit spatial structure to communicate the arrangement of image regions: LLaVA-UHD (Guo et al., 2024) partitions images into variable-sized slices and inserts row/column separators, while Molmo (Deitke et al., 2025) uses spatial tokens when arranging multi-crop features. Other complementary works enable true native or dynamic-resolution processing within the vision encoder. Qwen2-VL (Wang et al., 2024) replaces fixed positional embeddings with 2D RoPE, while Qwen2.5-VL (Bai et al., 2025b) trains a native dynamic-resolution ViT from scratch. More recent models such as Kimi-VL (Kimi Team et al., 2025), NativeRes-LLaVA (Niu et al., 2025), and Qwen3-VL (Bai et al., 2025a) continue this predominantly ViT-based direction. In contrast, hierarchical encoders such as ConvLLaVA (Ge et al., 2024) and FastVLM (Vasu et al., 2025) improve high-resolution efficiency using ConvNeXt and hybrid convolution-transformer backbones, respectively, but operate at predefined fixed high-resolution settings. We close this gap by adapting pretrained hierarchical convolutional and hybrid vision encoders to the native resolution of inputs during VLM instruction tuning, without image tiling, explicit layout tokens, or additional vision pretraining.

Coordinate representations for visual grounding. Generative grounding methods typically represent coordinates as either numerical text or quantized location tokens. Text-based approaches such as Shikra (Chen et al., 2023) and Qwen3-VL (Bai et al., 2025a) reuse the language vocabulary and can provide strong grounding accuracy, but require more output tokens. Quantized representations instead encode coordinates compactly using dedicated tokens, as in Kosmos-2 (Peng et al., 2023) and more recently Rex-Omni (Jiang et al., 2026). Recent methods such as GETok (Ren et al., 2026) and LocateAnything (Wang et al., 2026) further improve the tokenization or decoding of discrete coordinates. Complementary to these approaches, we study the training objective itself and introduce geometry-aware supervision for quantized coordinates, improving grounding without changing the underlying representation or adding inference-time computation.

Training-free confidence from location tokens. Generative grounding models do not naturally provide detector-style box confidence. Griffon and Griffon v2 (Zhan et al., 2024; 2025) derive training-free scores from generated coordinate and label likelihoods, while LMM-Det (Li et al., 2025c) finds simple probability aggregation insufficient and instead trains explicit confidence prediction. MTLA (Shalam et al., 2026) estimates confidence from localized attention within predicted regions. In contrast, we exploit the full predictive distribution of quantized location tokens, accumulating probability mass over alternative coordinates that preserve the generated box above a high IoU threshold. This yields a geometry- and scale-aware confidence score without additional training, supervision, model components, or repeated generation.

3 Method

FastVLM2 builds on the architecture used in FastVLM (Vasu et al., 2025), comprising a hierarchical vision encoder, a lightweight projector, and an autoregressive LLM. We illustrate the architecture in Figure 2 and provide details in Appx. A. FastVLM2 introduces three complementary components. First, we adapt pretrained hierarchical vision encoders to native-resolution inputs using only VLM fine-tuning, without additional vision pretraining. We validate this adaptation on two backbone families: a convolution-transformer hybrid (FastViTHD) and a convolutional architecture (ConvNeXt). Second, we improve quantized grounding with geometry-aware supervision during supervised fine-tuning (SFT). Third, we introduce a training-free box confidence estimator that operates directly on quantized location-token distributions and can be applied to any model using this representation.

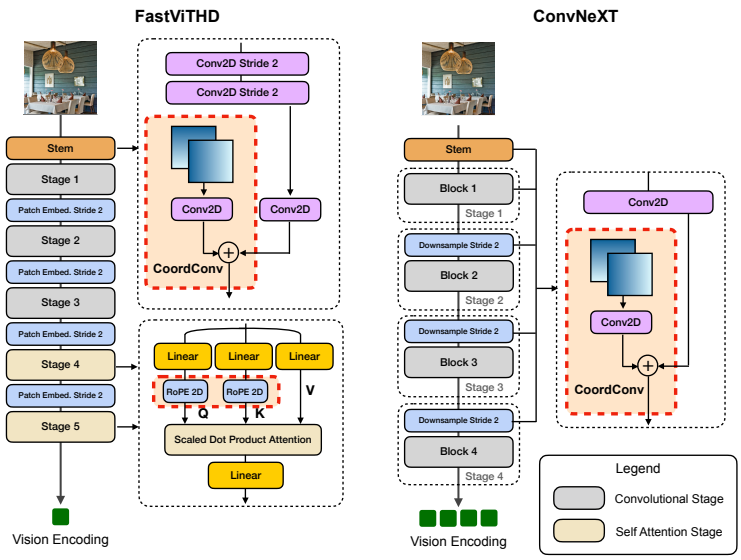


Figure 2: **Native-resolution adaptations for hierarchical vision encoders.** We illustrate the proposed architectural interventions for FastViTHD, a convolution-transformer hybrid, and ConvNeXt, a fully convolutional hierarchy. Proposed model interventions are highlighted with red dashed boxes.

3.1 Native-Resolution Hierarchical Vision Encoder

Naively exposing a fixed-resolution pretrained hierarchical encoder to arbitrary image sizes can degrade performance because the encoder has limited information about absolute spatial position as feature-map sizes vary. To support arbitrary image sizes and aspect ratios, we augment these encoders with explicit spatial information while preserving their architecture and initialization.

Spatial coordinate injection with CoordConv. Convolution captures local structure efficiently but does not explicitly encode absolute feature location. We inject two-dimensional coordinates using CoordConv (Liu et al., 2018) through a zero-initialized residual projection, so that the modified encoder initially preserves the pretrained representation and gradually learns to use absolute position during VLM fine-tuning. The placement of spatial encoding depends on the backbone architecture (Figure 2). For FastViTHD, a convolution-transformer hybrid, a single CoordConv injection at the convolutional stem is sufficient, while the subsequent self-attention stages use 2D RoPE (Heo et al., 2024). For ConvNeXt, which contains no attention stages, CoordConv is instead applied at the stem and downsampling blocks. These lightweight modifications require no additional vision pretraining and are learned entirely during VLM fine-tuning. The resulting encoders operate directly on native-resolution images and do not require image tiling or explicit layout tokens at inference time.

3.2 Improving Quantized Visual Grounding

Location tokenization and initialization. Following prior work on generative grounding (Jiang et al., 2026; Wang et al., 2026), we represent normalized coordinates with $K = 1000$ discrete location tokens, $\{\texttt{<loc_0>}, \ldots, \texttt{<loc_999>}\}$, so that each bounding box is generated using four tokens. We append these tokens to the base LLM vocabulary and initialize them by copying the last K pretrained token embeddings, which performs best in our ablations. VLM training proceeds in multiple stages (see Appx. D), with the location vocabulary introduced early using only the standard next-token prediction (NTP) objective, allowing the model to first learn the discrete coordinate representation before geometric supervision is incorporated in subsequent stages.

Geometry-aware auxiliary objectives. Standard NTP treats location tokens as independent categorical symbols and does not explicitly encode the geometry of the underlying coordinate space. We therefore explore several auxiliary objectives that inject geometric structure directly into the location-token predictions. *Distance-aware soft cross-entropy* replaces the one-hot coordinate target with a distance-decaying distribution centered on the ground-truth bin, assigning smaller penalties

Encoder	Patch Size	Image Proc.	Positional Encoding	Avg. 5
FastVLM2-FVHD	64	Fixed Res.	None	64.1
		Native	None	61.4
		Native	2D RoPE: Stage 4, 5	65.7
		Native	CoordConv: Stem	65.8
		Native	2D RoPE: Stage 4, 5 + CoordConv: Stem	**66.8**
	32	Native	2D RoPE: Stage 4, 5 + CoordConv: Stem	**73.5**
FastVLM2-CN	32	Fixed Res.	None	68.6
		Native	None	70.3
		Native	CoordConv: Stem	73.5
		Native	CoordConv: Stem + Downsample	**73.6**
		Native	CoordConv: Stem + Downsample + Blocks	73.3

(a) **Architectural interventions.** We ablate patch size and positional encoding choices for adapting hierarchical encoders to native-resolution inputs. Finer spatial grids improve accuracy at the cost of $4\times$ more visual tokens.

Stage 1.5	Stage 2.0	Avg. 5
Fixed Res.	Fixed Res.	**64.1**
Fixed Res.	Native	61.4
Native	Native	44.2

(b) **Training stage.** Keeping Stage 1.5 fixed avoids the severe degradation from introducing native resolution too early.

Scale	Rotation	Avg. 5
–	–	66.8
✓	–	68.3
✓	✓	**69.2**

(c) **Augmentation.** Scale and mild rotation augmentation improve native-resolution training by $+2.4$ points.

Table 1: **FastVLM2 native-resolution ablations.** FastViTHD (FVHD) and ConvNeXt (CN) are two hierarchical encoder instantiations of FastVLM2. We study architectural interventions, training curriculum, and data augmentation for native-resolution processing.

to nearby predictions. *Numeric coordinate supervision* applies an ℓ_1 loss to differentiable coordinate estimates obtained from the location-token distributions. At the box level, we ablate *GIoU* loss (Rezatofighi et al., 2019) between the differentiable predicted and ground-truth boxes, as well as an *ordering* loss that penalizes malformed boxes with $x_1 > x_2$ or $y_1 > y_2$. All auxiliary objectives operate directly on the LLM outputs and require no task-specific prediction heads or additional inference-time computation. Full definitions are provided in Appx. B, and their individual effects and combinations are evaluated in Section 4.2.

3.3 Training-Free Box Confidence

Discrete location tokens provide a distribution over candidate coordinates that can be used to estimate box confidence without additional training. Rather than measuring uncertainty only in token space, we score alternatives by their geometric effect on the predicted box. Let $\hat{B}$ denote the generated box and $B_{t\leftarrow k}$ the box obtained by replacing coordinate t with candidate token k while keeping the others fixed. We define

$$C_t^\tau = \sum_{k=0}^{K-1} p_t(k)\mathbf{1}\left[\mathrm{IoU}\left(B_{t\leftarrow k}, \hat{B}\right) \geq \tau\right], \tag{1}$$

where K is the number of discrete location tokens, $p_t(k)$ is the predicted probability of candidate token k for coordinate t, and τ controls the tolerated geometric deviation (we use $\tau = 0.9$). Thus, C_t^τ is the probability mass assigned to coordinate alternatives that preserve the predicted box at high IoU. We average the four coordinate confidences to obtain a box-level score, which is scale-aware by construction. Since only nearby bins can satisfy the high-IoU constraint, we evaluate this small feasible neighborhood, making the estimator efficient and requiring only logits already produced during decoding, with no additional parameters, forward passes, or attention outputs.

4 Experiments

4.1 Native-Resolution Adaptation Ablations

We first perform a series of ablations to establish the native-resolution training recipe before studying our full model. Unless otherwise specified, these experiments use a Qwen2.5-1.5B LLM decoder. We empirically find that ChartQA, OCRBench, DocVQA, InfoVQA, and AI2D are particularly sensitive to input resolution while exhibiting relatively low run-to-run variance, making them useful for isolating the effects of our design choices. To keep the main-paper ablations compact, we therefore report **Avg. 5**, the mean across these five benchmarks.

When to introduce native-resolution training. FastVLM (Vasu et al., 2025) training consists of three stages: Stage 1.0 for vision-language alignment, Stage 1.5 for resolution scaling, and Stage 2.0 for supervised fine-tuning. We first study when native-resolution training should be introduced, using the pretrained FastViTHD encoder with its default positional encoding. As shown in Table 1b, transitioning to native resolution only in Stage 2.0 reduces Avg. 5 from 64.1 to 61.4, indicating that the pretrained encoder does not directly generalize to variable resolutions and aspect ratios. Introducing native resolution earlier, during Stage 1.5, further degrades performance to 44.2. These results suggest that fixed-aspect-ratio resolution scaling in Stage 1.5 provides an important intermediate adaptation step before native-resolution fine-tuning. We therefore keep fixed-resolution scaling in Stage 1.5 and switch to native-resolution training in Stage 2.0 for all later experiments.

Architectural interventions for native resolution. We validate the design of lightweight architectural interventions using two substantially different hierarchical encoders: FastViTHD, which combines convolutional and self-attention stages, and ConvNeXt, which provides a purely convolutional hierarchy. For FastViTHD, Table 1a shows that naively applying native-resolution training with its default Conditional Positional Encoding (CPE) (Vasu et al., 2023) reaches an Avg. 5 of 61.4, compared with 64.1 for fixed-resolution training. Introducing 2D RoPE in the self-attention stages improves Avg. 5 to 65.7, while injecting CoordConv at the convolutional stem reaches 65.8. Combining both interventions provides complementary spatial cues to the convolutional and self-attention components and further improves Avg. 5 to 66.8.

We then apply the same FastVLM2 design principle to ConvNeXt. Native-resolution training alone reaches 70.3 Avg. 5 compared with 68.6 at fixed resolution. Adding CoordConv at the stem improves performance to 73.5, while extending coordinate injection to the downsampling layers gives the best result of 73.6. Injecting CoordConv into every convolutional block provides no further benefit. Together, these results show that FastVLM2 does not rely on a single encoder-specific mechanism: the appropriate positional intervention follows the structure of the hierarchy, with FastViTHD benefiting from CoordConv and 2D RoPE, and ConvNeXt requiring only coordinate injection.

Patch size provides an additional accuracy-efficiency trade-off. FastViTHD uses a patch size of 64 by default, whereas the ConvNeXt configuration uses a finer spatial grid corresponding to a patch size of 32. Reducing the FastViTHD patch size from 64 to 32, while retaining the same FastVLM2 positional interventions, increases Avg. 5 from 66.8 to 73.5. However, halving the patch size along both spatial dimensions produces $4\times$ more visual tokens at a fixed input resolution, increasing both LLM prefill latency and peak memory usage. Since our goal is to enable native-resolution processing while preserving the efficiency required for on-device deployment, we retain a patch size of 64 for FastViTHD in all subsequent experiments. Details of the patch-32 variant are provided in Appx. A.

Data augmentation for native-resolution training. Finally, we study data augmentation after establishing the FastVLM2 native-resolution adaptation recipe. We apply random scale augmentation while preserving aspect ratio, sampling a resize factor uniformly between 1.0 and 2.0. We only scale images above their original resolution to avoid removing text and other fine-grained visual details. As shown in Table 1c, scale augmentation improves Avg. 5 from 66.8 to 68.3. Adding mild rotation augmentation, with rotations sampled between 0° and 20°, further improves Avg. 5 to 69.2, yielding a 2.4-point gain over training without augmentation. All augmentation experiments use the FastViTHD configuration with 2D RoPE and CoordConv identified above.

4.2 Grounding Ablations

For grounding experiments, we augment the training mixture with additional grounding data and conduct all ablations using a Qwen2.5-3B LLM decoder. We first study how to initialize the new location-token embeddings, then investigate geometry-aware training objectives.

Location token initialization. Introducing $K = 1000$ location tokens expands the LLM vocabulary with embeddings that have not been pretrained. We therefore evaluate several initialization strategies. *Last-N* copies embeddings from the last K non-special tokens in the base vocabulary. *Numeric* initializes location tokens from existing digit-bearing tokens, while *Mean* assigns every location token the mean embedding of the original vocabulary. *Sinusoidal PE* constructs a sinusoidal embedding over the ordered coordinate bins and matches its scale to the pretrained embedding distribution. Finally, *Random* uses the default initialization when resizing the tokenizer vocabulary.

Grounding	Location Token Init.	Avg. 5	RefCOCO Avg.
Plain Text	–	69.4	64.0
Location Tokens	Last-N	68.2	**62.0**
	Sinusoidal	68.3	61.3
	Random	68.9	60.8
	Mean	68.7	57.8
	Numeric	**69.0**	57.0

(a) Location-token initialization.

Coord. Rep.	Auxiliary Objective	Avg. 5	RefCOCO Avg.
Text	–	73.3	77.3
Location Tokens	None (NTP only)	73.2	71.9
	DA-SoftCE (Laplacian)	73.0	76.2
	DA-SoftCE (Gaussian)	**73.5**	**76.4**
	ℓ_1 (coordinate)	72.9	74.7
	GIoU (box)	72.7	74.9

(b) Geometry-aware objectives.

Table 2: **Grounding ablations. Left:** comparison of location-token initialization strategies, where Last-N gives the strongest RefCOCO grounding accuracy among quantized variants. **Right:** comparison of geometry-aware auxiliary objectives, where Gaussian DA-SoftCE performs best among the quantized variants on both Avg. 5 and RefCOCO Avg.

Table 2a compares these choices with the plain-text coordinate representation. Among quantized representations, simply reusing the last K pretrained embeddings performs best on grounding, reaching a RefCOCO average of 62.0. More structured initializations, including numeric and sinusoidal embeddings, provide no additional grounding benefit. We therefore use Last-N initialization for all subsequent grounding experiments. Plain-text coordinates remain stronger at this stage, motivating the geometry-aware objectives studied next.

Geometry-aware losses for quantized grounding. We next study geometry-aware auxiliary objectives for quantized grounding. Location tokens are introduced during Stage 1.5 using only the standard next-token prediction (NTP) objective; auxiliary grounding losses are deferred to Stage 2.0 while retaining NTP throughout training. Our primary coordinate-level objective is *distance-aware soft cross-entropy* (DA-SoftCE), which replaces the one-hot target with a distance-decaying distribution over neighboring location tokens. We consider Gaussian and Laplacian kernels and sweep the kernel width over $\{1, 5, 10, 20\}$. We additionally ablate coordinate-wise ℓ_1 and box-level GIoU objectives.

As shown in Table 2b, Gaussian DA-SoftCE provides the strongest overall quantized-grounding result, improving RefCOCO Avg. from 71.9 with NTP alone to 76.4 while also obtaining the highest Avg. 5 among the quantized variants. We therefore use Gaussian DA-SoftCE as the sole auxiliary grounding objective in our final training recipe. Geometry-aware supervision thus recovers most of the grounding accuracy lost to quantization, without task-specific prediction heads or additional inference-time computation.

4.3 Training-Free Confidence Estimation

Grounding VLMs typically lack reliable per-box confidence scores, limiting the use of standard detection metrics such as AP. Prior work (Shalam et al., 2026) shows that this is particularly problematic for plain-text, digit-tokenized coordinates, where token likelihood depends strongly on decoding order and digit position and provides a weak localization-confidence signal. Quantized location tokens instead expose a single distribution over an ordered coordinate vocabulary for each box edge, which enables our geometry-aware confidence score.

Table 3 shows that this property transfers across three independently trained location-token models. Our box-space score improves mAP over the score-free protocol by $+8.52$, $+8.56$, and $+8.34$ for FastVLM2-1.8B, Rex-Omni, and LocateAnything, respectively, without retraining or model-specific fitting. This enables meaningful confidence-ranked AP evaluation for quantized grounding models.

We compare four ranking protocols. *Score-free* assigns every box the same confidence, as done by Rex-Omni, while *emission order* ranks boxes by their autoregressive generation order and serves as a free baseline. *Best bin-space* denotes the strongest confidence computed directly from the coordinate-token distribution, including probability-, entropy-, margin-, and distance-based scores. *Best box-space* instead evaluates these distributions in the geometry of the predicted box, producing a scale-aware confidence estimate. Box-space ranking is consistently strongest across all three models. Additional scoring variants, thresholds, and ablations are provided in Appx. C.

Confidence	FastVLM2-1.8B		Rex-Omni-3B		LocateAnything-3B	
Score	mAP	$F1_{50}$	mAP	$F1_{50}$	mAP	$F1_{50}$
Score-free	36.05	71.54	35.64	72.01	34.56	68.79
Emission order	38.08	72.48	37.13	72.30	36.19	69.16
Best bin-space	41.20	72.71	39.56	72.40	40.47	69.93
Best box-space	**44.57**	**73.14**	**44.20**	**72.89**	**42.90**	**70.39**
Δ over score-free	**+8.52**	+1.60	**+8.56**	+0.88	**+8.34**	+1.60

Table 3: **Training-free confidence estimation.** Our box-space scoring mechanism generalizes across models using quantized location tokens and requires no additional training, improving mAP by more than 8 points over the score-free baseline for all evaluated models.

Models	Vision Encoder		Prefill Latency (s) ↓	Avg. VQA Perf. ↑	Avg. Refer. Exp. ↑	Avg. Dense Grounding ↑
	Name	Size (M) ↓				
InternVL3.5-1B	InternViT	304	3.1	68.2	51.7	30.6
FastVLM2-0.7B	FastViTHD	**125**	**0.4**	**73.4**	**77.0**	**51.0**
MiniCPM-V-4.6-1.3B	SigLIP2	418	1.1	72.7	–	–
InternVL3.5-2B	InternViT	304	4.6	74.6	52.1	33.1
Qwen3-VL-2B	QwenViT	306	3.6	77.3	69.0	36.2
Ovis2.5-2B	SigLIP2	414	4.6	**81.2**	65.7	38.1
FastVLM2-1.8B	FastViTHD	**125**	**0.6**	77.9	**78.6**	**52.3**
Rex-Omni-3B	ViT	632	6.3	–	75.9	49.9
LocateAnything-3B	MoonViT	417	8.2	–	77.7	52.7
InternVL3.5-4B	InternViT	304	8.6	77.9	63.1	41.4
Molmo2-4B	SigLIP2	383	–	75.0	–	–
Qwen3-VL-4B (instruct)	QwenViT	306	5.6	**82.1**	73.5	44.8
FastVLM2-4B	FastViTHD	**125**	**1.2**	81.7	**81.4**	**52.9**

Table 4: **FastVLM2 improves the accuracy-latency trade-off for VQA, referring-expression comprehension, and dense grounding.** Each average is an arithmetic mean. Avg. VQA Perf.: ChartQA, OCRBench, TextVQA, DocVQA, InfoVQA, AI2D, RealWorldQA, and SEED-Bench. Avg. Refer. Exp.: F1@mIoU on RefCOCOg val, RefCOCOg test, and HumanRef. Avg. Dense Grounding: F1@IoU Mean on COCO and LVIS. **Bold** indicate the best results.

4.4 Main Results

Training setup. FastVLM2 is trained with supervised fine-tuning (SFT) followed by reinforcement learning (RL). We use the FastViTHD vision encoder (Vasu et al., 2023) and Qwen3 LLM decoders (Yang et al., 2025a) of different sizes in all of our main results. SFT proceeds in five stages, detailed in Appx. D, on a mixture of datasets spanning general VQA, document/chart understanding, captioning, and dense grounding. The RL stage then optimizes the full SFT model with GSPO (Zheng et al., 2025) on a mixture of grounding and general instruction-following data: the reward in the grounding tasks is based on an IoU-based F_1 score, while the reward in other tasks is obtained by a frozen Qwen3-32B model acting as judge (Appx. D.5). Details about both SFT and RL training datasets are provided in Appx. D.1. All training ablations are performed on SFT-only models; all other results use the final model after both SFT and RL, unless marked *SFT*.

Evaluation setup. We evaluate FastVLM2 on eight standard VLM benchmarks (AI2D, SEED-Bench, RealWorldQA, TextVQA, ChartQA, DocVQA, InfoVQA, and OCRBench) using `lmms-eval` v0.7.1, and on grounding benchmarks spanning referring-expression comprehension (RefCOCO, RefCOCO+, RefCOCOg, and HumanRef) and dense detection (COCO and LVIS). Full benchmark and metric definitions are given in Appx. E. We measure the latency of all models on the same hardware (Apple MacBook M1 Max, 32 GB) for a consistent comparison (Appx. G).

Baselines. Following prior work (Jiang et al., 2026; Wang et al., 2026), we compare against three categories of methods: (1) *generalist vision-language models* (e.g., InternVL3.5, MiniCPM-V, Qwen3-VL, Ovis2.5, Molmo2), which are trained and evaluated on both standard VLM benchmarks and grounding; (2) *grounding-specialist vision-language models* (Rex-Omni, LocateAnything), which are trained and evaluated only on localization tasks; and (3) *specialized object detectors* (Grounding DINO, DETR, Deformable-DETR, DINO), included as closed- and open-set reference points rather than direct competitors.

Method	Encoder Size (M) ↓	Prefill Latency (s) ↓	Overall Latency (s) ↓	COCO mAP ↑	COCO (F1@IoU) ↑			LVIS (F1@IoU) ↑		
					0.5	0.95	Mean	0.5	0.95	Mean
Open-set Specialized Detectors										
Grounding DINO-Swin-T	–	–	–	51.5	69.8	23.0	56.6	47.7	22.7	38.8
Closed-set Specialized Detectors										
DETR-R50	–	–	–	41.5	65.9	13.6	48.3	–	–	–
Deformable-DETR-R50	–	–	–	49.4	69.7	17.7	54.7	–	–	–
DINO-R50	–	–	–	51.7	68.8	21.1	55.6	–	–	–
DINO-Swin-L	–	–	–	59.6	**75.6**	**25.4**	**62.1**	–	–	–
Grounding Specialist Vision-Language Models										
Rex-Omni-3B	632	1.5	2.0	–	72.0	15.9	52.9	**64.3**	20.7	46.9
LocateAnything-3B	417	1.7	–	–	70.1	**19.3**	54.7	62.3	**31.1**	**50.7**
Generalist Vision-Language Models										
Qwen3-VL-2B	306	0.8	1.6	–	48.2	11.5	35.6	50.1	17.5	36.7
InternVL3.5-1B	304	0.9	1.6	–	51.5	8.0	34.4	42.1	9.7	26.7
InternVL3.5-2B	304	1.5	2.4	–	53.4	9.7	37.1	44.1	13.7	29.1
Ovis2.5-2B	414	0.9	2.4	–	56.2	10.3	38.7	54.4	15.8	37.4
InternVL3.5-4B	304	2.7	4.3	–	60.5	14.2	44.9	51.4	19.4	37.9
Qwen3-VL-4B	306	1.3	2.8	–	63.0	14.2	46.1	59.8	20.0	43.5
FastVLM2-0.7B	**125**	**0.1**	**0.3**	44.3	73.1	17.9	55.4	60.9	23.2	46.6
FastVLM2-1.8B	**125**	**0.2**	**0.5**	44.9	**73.6**	<u>19.0</u>	**56.2**	62.9	24.1	48.3
FastVLM2-4B	**125**	**0.4**	**0.8**	44.5	**73.6**	18.7	<u>56.1</u>	<u>64.0</u>	<u>24.4</u>	<u>49.7</u>

Table 5: **Dense grounding performance on COCO and LVIS.** Prefill latency, overall latency, and vision-encoder size are reported alongside grounding accuracy to characterize the efficiency-accuracy trade-off (Appx. G). "–" denotes results that are unavailable, not reported, or not applicable. **Bold** and <u>underline</u> indicate the best and second-best results, respectively.

Results. Table 4 reports, for each model, the average VQA performance, average referring-expression performance, and average dense grounding performance, together with prefill latency and vision-encoder size. FastVLM2-4B is competitive with Qwen3-VL-4B on average VQA performance (81.7 vs. 82.1) while using a vision encoder $2.4\times$ smaller (125M vs. 306M parameters) and $4.7\times$ lower prefill latency (1.2 s vs. 5.6 s), and it clearly exceeds Qwen3-VL-4B on both average referring expression (81.4 vs. 73.5) and average dense grounding (52.9 vs. 44.8). FastVLM2-4B also exceeds the grounding specialists Rex-Omni-3B and LocateAnything-3B on average referring expression (81.4 vs. 75.9 and 77.7) and matches or exceeds them on average dense grounding (52.9 vs. 49.9 and 52.7), while additionally supporting general VQA (81.7 average) and using a 3.3–$5.1\times$ smaller vision encoder. Note that dense grounding results use our training-free box-space confidence score to rank detections. The same efficiency trend holds at smaller scale: FastVLM2-1.8B matches InternVL3.5-4B on average VQA performance (77.9 vs. 77.9) with $14.3\times$ lower prefill latency, and even FastVLM2-0.7B's average referring-expression performance (77.0) already exceeds Rex-Omni-3B's.

Per-benchmark numbers underlying these averages are given in Tables 14 and 15 (Appx. F) for VQA and referring expression, and in Table 5 for dense grounding. Table 4 uses each model's default settings, whereas Figure 1 sweeps the maximum input size (Appx. G), so the two differ slightly and should not be compared directly. Beyond accuracy, Figure 1 shows that these gains carry over directly to efficiency: FastVLM2-4B reaches VQA accuracy comparable to Qwen3-VL-4B at $4.6\times$ lower prefill latency, while FastVLM2-1.8B exceeds Rex-Omni's grounding accuracy at $7.3\times$ lower end-to-end latency. Together, FastVLM2 pairs broad VLM and dense-grounding capability with higher efficiency, driven by native-resolution encoding and compact location tokens.

5 Conclusion

We introduced FastVLM2, a unified approach to VLM efficiency at both ends of the pipeline. We show that pretrained hierarchical vision encoders can be adapted to native-resolution inputs during VLM instruction tuning, improving input-side efficiency without additional vision pretraining or image tiling. For grounding, geometry-aware supervision substantially improves existing quantized location tokens, closing much of the gap to textual coordinates while preserving their shorter outputs. These location tokens also support training-free, geometry-aware confidence estimation that transfers across independently trained grounding models. Together, these results provide a practical recipe for efficient visual encoding and structured prediction in resource-constrained VLMs.

AI USE STATEMENT

In this work, we used generative AI tools to assist with implementing methods, translating, cleaning and reformatting datasets, supporting qualitative and thematic data analysis, and interpreting results. We did not use generative AI tools to generate synthetic datasets, propose or refine hypotheses, design or provide feedback on research methodology or experiments. Additionally, the following use cases were not applicable to our work: developing theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, or assisting in writing proofs. We also used generative AI tools to create or modify scientific figures and images, edit the paper to improve readability, and propose a title and keywords.

We reviewed all AI-assisted work and manually verified and tested LLM-generated code and artifacts for correctness. We take full responsibility for the final content of this work, including all text, claims, and artifacts produced with the assistance of generative AI.

REPRODUCIBILITY STATEMENT

We describe our native-resolution architecture and grounding objectives in Section 3, with complete training configurations, per-stage hyperparameters, and the training data mixture detailed in Appx. D. The formulations and hyperparameter sweeps for the geometry-aware grounding losses and training-free box-confidence estimation are provided in Appx. B and Appx. C, respectively. Details of the reinforcement-learning stage, including the training data, reward details and optimization hyperparameters are provided in Appx. D.5. Finally, all benchmarks are evaluated under a single shared harness, with exact task names and metric keys listed in Appx. E.

REFERENCES

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025a.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025b.

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

Kai Chen, Yunhao Gou, Runhui Huang, Zhili Liu, Daxin Tan, Jing Xu, Chunwei Wang, Yi Zhu, Yihan Zeng, Kuo Yang, et al. Emova: Empowering language models to see, hear and speak with vivid emotions. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Keqin Chen, Zhao Zhang, Weili Zeng, Richong Zhang, Feng Zhu, and Rui Zhao. Shikra: Unleashing multimodal llm's referential dialogue magic. *arXiv preprint arXiv:2306.15195*, 2023.

Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, et al. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 24185–24198, 2024.

Christopher Clark, Jieyu Zhang, Zixian Ma, Jae Sung Park, Rohun Tripathi, Sangho Lee, Mohammadreza Salehi, Jason Ren, Chris Dongjoo Kim, Yinuo Yang, et al. Molmo2: Open weights and data for vision-language models with video understanding and grounding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 28652–28668, 2026.

Mostafa Dehghani, Basil Mustafa, Josip Djolonga, Jonathan Heek, Matthias Minderer, Mathilde Caron, Andreas Steiner, Joan Puigcerver, Robert Geirhos, Ibrahim Alabdulmohsin, Avital Oliver, Piotr Padlewski, Alexey Gritsenko, Mario Lucić, and Neil Houlsby. Patch n' pack: NaViT, a vision transformer for any aspect ratio and resolution. In *Proceedings of the Conference on Neural Information Processing Systems*, 2023.

Matt Deitke, Christopher Clark, Sangho Lee, Rohun Tripathi, Yue Yang, Jae Sung Park, Mohammadreza Salehi, Niklas Muennighoff, Kyle Lo, Luca Soldaini, et al. Molmo and pixmo: Open weights and open data for state-of-the-art vision-language models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 91–104. IEEE, 2025.

Fartash Faghri, Pavan Kumar Anasosalu Vasu, Cem Koc, Vaishaal Shankar, Alexander T Toshev, Oncel Tuzel, and Hadi Pouransari. MobileCLIP2: Improving multi-modal reinforced training. *Transactions on Machine Learning Research*, 2025. Featured Certification.

Chunjiang Ge, Sijie Cheng, Ziming Wang, Jiale Yuan, Yuan Gao, Jun Song, Shiji Song, Gao Huang, and Bo Zheng. Convllava: Hierarchical backbones as visual encoder for large multimodal models. *arXiv preprint arXiv:2405.15738*, 2024.

Jiawei Guo, Tianyu Zheng, Yizhi Li, Yuelin Bai, Bo Li, Yubo Wang, King Zhu, Graham Neubig, Wenhu Chen, and Xiang Yue. Mammoth-vl: Eliciting multimodal reasoning with instruction tuning at scale. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13869–13920, 2025.

Zonghao Guo, Ruyi Xu, Yuan Yao, Junbo Cui, Zanlin Ni, Chunjiang Ge, Tat-Seng Chua, Zhiyuan Liu, and Gao Huang. LLaVA-UHD: an lmm perceiving any aspect ratio and high-resolution images. In *Proceedings of the European Conference on Computer Vision*, 2024.

Agrim Gupta, Piotr Dollár, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5351–5359, 2019. doi: 10.1109/CVPR.2019.00550.

Byeongho Heo, Song Park, Dongyoon Han, and Sangdoo Yun. Rotary position embedding for vision transformer. In *Proceedings of the European Conference on Computer Vision*, 2024.

Mengzhao Jia, Wenhao Yu, Kaixin Ma, Tianqing Fang, Zhihan Zhang, Siru Ouyang, Hongming Zhang, Dong Yu, and Meng Jiang. Leopard: A vision language model for text-rich multi- image tasks. *Trans. Mach. Learn. Res.*, 2025, 2025. URL `https://openreview.net/forum?id=R2rasAEPVi`.

Qing Jiang, Xingyu Chen, Zhaoyang Zeng, Junzhi Yu, and Lei Zhang. Rex-Thinker: Grounded object referring via chain-of-thought reasoning. *arXiv preprint arXiv:2506.04034*, 2025a.

Qing Jiang, Lin Wu, Zhaoyang Zeng, Tianhe Ren, Yuda Xiong, Yihao Chen, Liu Qin, and Lei Zhang. Referring to any person. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 21667–21678. IEEE, 2025b.

Qing Jiang, Junan Huo, Xingyu Chen, Yuda Xiong, Zhaoyang Zeng, Yihao Chen, Tianhe Ren, Junzhi Yu, and Lei Zhang. Detect anything via next point prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026.

Sahar Kazemzadeh, Vicente Ordonez, Mark Matten, and Tamara Berg. Referitgame: Referring to objects in photographs of natural scenes. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014.

Aniruddha Kembhavi, Mike Salvato, Eric Kolve, Minjoon Seo, Hannaneh Hajishirzi, and Ali Farhadi. A diagram is worth a dozen images. In *European conference on computer vision*, pp. 235–251. Springer, 2016.

Kimi Team, Angang Du, Bohong Yin, Bowei Xing, Bowen Qu, Bowen Wang, Cheng Chen, Chenlin Zhang, Chenzhuang Du, Chu Wei, et al. Kimi-vl technical report. *arXiv preprint arXiv:2504.07491*, 2025.

Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, Tom Duerig, and Vittorio Ferrari. The open images dataset V4: Unified image classification, object detection, and visual relationship detection at scale. *International Journal of Computer Vision (IJCV)*, 2020.

Bo Li, Hao Zhang, Kaichen Zhang, Dong Guo, Yuanhan Zhang, Renrui Zhang, Feng Li, Ziwei Liu, and Chunyuan Li. Llava-next: What else influences visual instruction tuning beyond data?, May 2024a. URL https://llava-vl.github.io/blog/2024-05-25-llava-next-ablations/.

Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, and Chunyuan Li. Llava-onevision: Easy visual task transfer. *Trans. Mach. Learn. Res.*, 2025, 2025a. URL https://openreview.net/forum?id=zKv8qULV6n.

Bohao Li, Rui Wang, Guangzhi Wang, Yuying Ge, Yixiao Ge, and Ying Shan. Seed-bench: Benchmarking multimodal llms with generative comprehension, 2023. URL https://arxiv.org/abs/2307.16125.

Hongyu Li, Jinyu Chen, Ziyu Wei, Shaofei Huang, Tianrui Hui, Jialin Gao, Xiaoming Wei, and Si Liu. Llava-st: A multimodal large language model for fine-grained spatial-temporal understanding. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025b.

Jincheng Li, Chunyu Xie, Ji Ao, Dawei Leng, and Yuhui Yin. Lmm-det: Make large multimodal models excel in object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025c.

Xiao Li, Yining Liu, Na Dong, Sitian Qin, and Xiaolin Hu. PartImageNet++ dataset: Scaling up part-based models for robust recognition. In *European Conference on Computer Vision (ECCV)*, 2024b.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision (ECCV)*, 2014.

Ziyi Lin, Chris Liu, Renrui Zhang, Peng Gao, Longtian Qiu, Han Xiao, Han Qiu, Chen Lin, Wenqi Shao, Keqin Chen, Jiaming Han, Siyuan Huang, Yichi Zhang, Xuming He, Hongsheng Li, and Yu Qiao. Sphinx: A mixer of weights, visual embeddings and image scales for multi-modal large language models. In *Proceedings of the European Conference on Computer Vision*, 2024.

Chang Liu, Henghui Ding, and Xudong Jiang. Gres: Generalized referring expression segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023a.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 26286–26296. IEEE, 2024a.

Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. Llava-next: Improved reasoning, ocr, and world knowledge, January 2024b. URL https://llava-vl.github.io/blog/2024-01-30-llava-next/.

Rosanne Liu, Joel Lehman, Piero Molino, Felipe Petroski Such, Eric Frank, Alex Sergeev, and Jason Yosinski. An intriguing failing of convolutional neural networks and the coordconv solution. In *Proceedings of the Conference on Neural Information Processing Systems*, 2018.

Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023b.

Yuliang Liu, Zhang Li, Mingxin Huang, Biao Yang, Wenwen Yu, Chunyuan Li, Xucheng Yin, Cheng lin Liu, Lianwen Jin, and Xiang Bai. Ocrbench: On the hidden mystery of ocr in large multimodal models, 2024c. URL https://arxiv.org/abs/2305.07895.

Shiyin Lu, Yang Li, Yu Xia, Yuwei Hu, Shanshan Zhao, Yanqing Ma, Zhichao Wei, Yinglun Li, Lunhao Duan, Jianshan Zhao, Yuxuan Han, Haijun Li, Wanying Chen, Junke Tang, Chengkun Hou, Zhixing Du, Tianli Zhou, Wenjie Zhang, Huping Ding, Jiahe Li, Wen Li, Gui Hu, Yiliang Gu, Siran Yang, Jiamang Wang, Hailong Sun, Yibo Wang, Hui Sun, Jinlong Huang, Yuping He, Shengze Shi, Weihong Zhang, Guodong Zheng, Junpeng Jiang, Sensen Gao, Yi-Feng Wu, Sijia Chen, Yuhui Chen, Qing-Guo Chen, Zhao Xu, Weihua Luo, and Kaifu Zhang. Ovis2.5 technical report, 2025. URL https://arxiv.org/abs/2508.11737.

Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan L Yuille, and Kevin Murphy. Generation and comprehension of unambiguous object descriptions. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Ahmed Masry, Jia Qing Tan, Shafiq Joty, Enamul Hoque, et al. Chartqa: A benchmark for question answering about charts with visual and logical reasoning. In *Findings of the association for computational linguistics: ACL 2022*, pp. 2263–2279, 2022.

Minesh Mathew, Dimosthenis Karatzas, and CV Jawahar. Docvqa: A dataset for vqa on document images. In *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 2199–2208. IEEE, 2021.

Minesh Mathew, Viraj Bagal, Rubèn Tito, Dimosthenis Karatzas, Ernest Valveny, and CV Jawahar. Infographicvqa. In *2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 2582–2591. IEEE, 2022.

Anindya Mondal, Sauradip Nag, and Anjan Dutta. Abacus: Adapting unified foundation model for bridging image count understanding and generation. *arXiv preprint arXiv:2606.23835*, 2026.

Junbo Niu, Yuanhong Zheng, Ziyang Miao, Hejun Dong, Chunjiang Ge, Hao Liang, Ma Lu, Bohan Zeng, Qiahao Zheng, Conghui He, and Wentao Zhang. Native visual understanding: Resolving resolution dilemmas in vision-language models. *arXiv preprint arXiv:2506.12776*, 2025.

Zhiliang Peng, Wenhui Wang, Li Dong, Yaru Hao, Shaohan Huang, Shuming Ma, and Furu Wei. Kosmos-2: Grounding multimodal large language models to the world. *arXiv preprint arXiv:2306.14824*, 2023.

Jordi Pont-Tuset, Jasper Uijlings, Soravit Changpinyo, Radu Soricut, and Vittorio Ferrari. Connecting vision and language with localized narratives. In *European conference on computer vision*, pp. 647–664. Springer, 2020.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

Xiangxuan Ren, Zhongdao Wang, Liping Hou, Pin Tang, Guoqing Wang, and Chao Ma. Grounding everything in tokens for multimodal large language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio 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 (CVPR)*, 2019.

Gabriel Sarch, Linrong Cai, Qunzhong Wang, Haoyang Wu, Danqi Chen, and Zhuang Liu. Vero: An open rl recipe for general visual reasoning. *arXiv preprint arXiv:2604.04917*, 2026.

Daniel Shalam, Emanuel Ben Baruch, Avi Ben Cohen, and Tal Remez. Propose and attend: Training-free mllm grounding confidence via multi-token localized attention. *arXiv preprint arXiv:2607.05978*, 2026.

Shuai Shao, Zeming Li, Tianyuan Zhang, Chao Peng, Gang Yu, Xiangyu Zhang, Jing Li, and Jian Sun. Objects365: A large-scale, high-quality dataset for object detection. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

Min Shi, Fuxiao Liu, Shihao Wang, Shijia Liao, Subhashree Radhakrishnan, Yilin Zhao, De-An Huang, Hongxu Yin, Karan Sapra, Yaser Yacoob, et al. Eagle: Exploring the design space for multimodal llms with mixture of encoders. In *International Conference on Learning Representations*, volume 2025, pp. 92628–92646, 2025.

Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. Towards vqa models that can read. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8309–8318. IEEE, 2019.

Hamed Rahimi Sujet AI, Allaa Boutaleb. Sujet-finance-qa-vision-100k: A large-scale dataset for financial document vqa, 2024. URL `https://huggingface.co/datasets/sujet-ai/Sujet-Finance-QA-Vision-100k`.

Yingbo Tang, Lingfeng Zhang, Shuyi Zhang, Yi-Nuo Zhao, and Xiaoshuai Hao. Roboafford: A dataset and benchmark for enhancing object and spatial affordance learning in robot manipulation. *Proceedings of the 33rd ACM International Conference on Multimedia*, 2025.

Shengbang Tong, Ellis Brown, Penghao Wu, Sanghyun Woo, Manoj Middepogu, Sai C Akula, Jihan Yang, Shusheng Yang, Adithya Iyer, Xichen Pan, et al. Cambrian-1: A fully open, vision-centric exploration of multimodal llms. *Advances in Neural Information Processing Systems*, 37:87310–87356, 2024.

Michael Tschannen, Alexey Gritsenko, Xiao Wang, Muhammad Ferjad Naeem, Ibrahim Alabdulmohsin, Nikhil Parthasarathy, Talfan Evans, Lucas Beyer, Ye Xia, Basil Mustafa, Olivier Hénaff, Jeremiah Harmsen, Andreas Steiner, and Xiaohua Zhai. Siglip 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features. *arXiv preprint arXiv:2502.14786*, 2025.

Dmitry Ustalov, Nikita Pavlichenko, Sergey Koshelev, Daniil Likhobaba, and Alisa Smirnova. Toloka visual question answering benchmark. *arXiv preprint arXiv:2309.16511*, 2023.

Pavan Kumar Anasosalu Vasu, James Gabriel, Jeff Zhu, Oncel Tuzel, and Anurag Ranjan. Fastvit: A fast hybrid vision transformer using structural reparameterization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

Pavan Kumar Anasosalu Vasu, Hadi Pouransari, Fartash Faghri, Raviteja Vemulapalli, and Oncel Tuzel. Mobileclip: Fast image-text models through multi-modal reinforced training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

Pavan Kumar Anasosalu Vasu, Fartash Faghri, Chun-Liang Li, Cem Koc, Nate True, Albert Antony, Gokul Santhanam, James Gabriel, Peter Grasch, Oncel Tuzel, and Hadi Pouransari. Fastvlm: Efficient vision encoding for vision language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

Jiaqi Wang, Pan Zhang, Tao Chu, Yuhang Cao, Yujie Zhou, Tong Wu, Bin Wang, Conghui He, and Dahua Lin. V3Det: Vast vocabulary visual detection dataset. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.

Shihao Wang, Shilong Liu, Yuanguo Kuang, Xinyu Wei, Yangzhou Liu, Zhiqi Li, Yunze Man, Guo Chen, Andrew Tao, Guilin Liu, Jan Kautz, Lei Zhang, and Zhiding Yu. Locateanything: Fast and high-quality vision-language grounding with parallel box decoding. In *Proceedings of the European Conference on Computer Vision*, 2026.

Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, Zhaokai Wang, Zhe Chen, Hongjie Zhang, Ganlin Yang, Haomin Wang, Qi Wei, Jinhui Yin, Wenhao Li, Erfei Cui, Guanzhou Chen, Zichen Ding, Changyao Tian, Zhenyu Wu, Jingjing Xie, Zehao Li, Bowen Yang, Yuchen Duan, Xuehui Wang, Zhi Hou, Haoran Hao, Tianyi Zhang, Songze Li, Xiangyu Zhao, Haodong Duan, Nianchen Deng, Bin Fu, Yinan He, Yi Wang, Conghui He, Botian Shi, Junjun He, Yingtong Xiong, Han Lv, Lijun Wu, Wenqi Shao, Kaipeng Zhang, Huipeng Deng, Biqing Qi, Jiaye Ge, Qipeng Guo, Wenwei Zhang, Songyang Zhang, Maosong Cao, Junyao Lin, Kexian Tang, Jianfei Gao, Haian Huang, Yuzhe Gu, Chengqi Lyu, Huanze Tang, Rui Wang, Haijun Lv, Wanli Ouyang, Limin Wang, Min Dou, Xizhou Zhu, Tong Lu, Dahua Lin, Jifeng Dai, Weijie Su, Bowen Zhou, Kai Chen, Yu Qiao, Wenhai Wang, and Gen Luo. Internvl3.5: Advancing open-source multimodal models in versatility, reasoning, and efficiency, 2025. URL https://arxiv.org/abs/2508.18265.

Chi Xie, Zhao Zhang, Yixuan Wu, Feng Zhu, Rui Zhao, and Shuang Liang. Described object detection: Liberating object detection with flexible expressions. In *Thirty-seventh Conference on Neural Information Processing Systems (NeurIPS)*, 2023.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025a.

Yue Yang, Ajay Patel, Matt Deitke, Tanmay Gupta, Luca Weihs, Andrew Head, Mark Yatskar, Chris Callison-Burch, Ranjay Krishna, Aniruddha Kembhavi, et al. Scaling text-rich image understanding via code-guided synthetic multimodal data generation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 17486–17505, 2025b.

Tianyu Yu, Zefan Wang, Chongyi Wang, Fuwei Huang, Wenshuo Ma, Zhihui He, Tianchi Cai, Weize Chen, Yuxiang Huang, Yuanqian Zhao, Bokai Xu, Junbo Cui, Yingjing Xu, Liqing Ruan, Luoyuan Zhang, Hanyu Liu, Jingkun Tang, Hongyuan Liu, Qining Guo, Wenhao Hu, Bingxiang He, Jie Zhou, Jie Cai, Ji Qi, Zonghao Guo, Chi Chen, Guoyang Zeng, Yuxuan Li, Ganqu Cui, Ning Ding, Xu Han, Yuan Yao, Zhiyuan Liu, and Maosong Sun. Minicpm-v 4.5: Cooking efficient mllms via architecture, data, and training recipe, 2025. URL https://arxiv.org/abs/2509.18154.

Yufei Zhan, Yousong Zhu, Zhiyang Chen, Fan Yang, Ming Tang, and Jinqiao Wang. Griffon: Spelling out all object locations at any granularity with large language models. In *Proceedings of the European Conference on Computer Vision*, 2024.

Yufei Zhan, Shurong Zheng, Yousong Zhu, Hongyin Zhao, Fan Yang, Ming Tang, and Jinqiao Wang. Griffon v2: Advancing multimodal perception with high-resolution scaling and visual-language co-referring. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025.

Hao Zhang, Feng Li, Shilong Liu, Lei Zhang, Hang Su, Jun Zhu, Lionel Ni, and Heung-Yeung Shum. DINO: DETR with improved denoising anchor boxes for end-to-end object detection. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=3mRwyG5one.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bo-Wen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, Jingren Zhou, and Junyang Lin. Group sequence policy optimization. *ArXiv*, abs/2507.18071, 2025. URL https://api.semanticscholar.org/CorpusID:280017753.

Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable {detr}: Deformable transformers for end-to-end object detection. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=gZ9hCDWe6ke.

A Architecture Details

FastVLM follows a standard vision-language architecture in which a pretrained vision encoder extracts spatial image features, a vision-language projector maps these features into the embedding space of the language model, and the resulting visual tokens are provided together with text tokens to a decoder-only LLM (Vasu et al., 2025). In this work, we retain this overall design and focus on adapting the pretrained hierarchical vision encoder to operate directly on images with varying resolutions and aspect ratios.

Figure 2 illustrates our architectural interventions for two representative hierarchical encoder families. For **FastViTHD**, a convolution-transformer hybrid, we inject explicit spatial information using CoordConv at the convolutional stem and apply 2D rotary positional embeddings (RoPE) within the self-attention stages. For **ConvNeXt**, which is fully convolutional, we apply CoordConv at the stem—the initial 4×4, stride-4 Conv2D layer preceding LayerNorm—and additionally to the Conv2D layers in each downsampling block. This configuration provides the strongest native-resolution performance among the ConvNeXt variants we evaluate. Further details and ablations on the choice and placement of these interventions are presented in Section 4.1, particularly Table 1a. Since both encoders are initialized from CLIP-pretrained models, we extract the spatial feature maps before the final global spatial pooling and embedding projection, retaining the dense spatial representation required by the VLM.

Increasing spatial resolution in FastViTHD. FastViTHD has an effective patch size of 64, compared with 32 for ConvNeXt. While the larger patch size reduces the number of visual tokens passed to the language model, it can limit spatial granularity for low-resolution images and tasks requiring fine-grained understanding, such as OCRBench, TextVQA, and RealWorldQA. To study this trade-off, we modify the patch embedding layer preceding Stage 5 and train a variant with stride 1, thereby preserving a higher spatial resolution in the final encoder stage. As shown in Table 1a, this modification improves Avg. 5 performance from 66.8 to 73.5. The increased spatial resolution, however, produces $4\times$ more visual tokens, matching the output token density of ConvNeXt, as illustrated in Figure 2. This variant therefore provides an additional accuracy-efficiency operating point, trading increased visual-token and prefill cost for finer spatial representation.

B Geometry-Aware Auxiliary Objectives

This section provides implementation details for the geometry-aware objectives used to train quantized location tokens. Location tokens are introduced during Stage 1.5 using standard next-token prediction (NTP). Geometry-aware objectives are enabled only during Stage 2.0, while NTP supervision is retained throughout training.

Let the coordinate vocabulary contain $K = 1000$ bins. For a supervised location-token position i, let $k_i \in \{0, \dots, K-1\}$ denote the target bin and $v_i = k_i/(K-1)$ its normalized coordinate. Given logits z_i, we define the distribution over the location-token vocabulary as

$$q_{i,j} = \frac{\exp(z_{i,s+j})}{\sum_{r=0}^{K-1} \exp(z_{i,s+r})}, \qquad j = 0, \dots, K-1, \tag{2}$$

where s denotes the vocabulary index of the first location token.

B.1 Distance-Aware Soft Cross-Entropy

Standard cross-entropy treats coordinate bins as independent classes and therefore does not distinguish between near and distant localization errors. We instead replace the one-hot target at location-token positions with a distance-aware target distribution. For target bin k, we consider Gaussian and Laplacian kernels,

$$\widetilde{T}_{k,j} = \begin{cases} \exp\left(-\dfrac{(k-j)^2}{2w^2}\right), & \text{Gaussian,} \\ \exp\left(-\dfrac{|k-j|}{w}\right), & \text{Laplacian,} \end{cases} \qquad T_{k,j} = \frac{\widetilde{T}_{k,j}}{\sum_{r=0}^{K-1} \widetilde{T}_{k,r}}, \tag{3}$$

where w controls the width of the soft target in coordinate-bin space. The distance-aware soft cross-entropy is

$$\ell_i^{\text{soft}} = -\sum_{j=0}^{K-1} T_{k_i,j} \log p_{i,s+j}, \tag{4}$$

where p_i is the softmax over the full language-model vocabulary. We mix this term with the standard hard-target cross-entropy,

$$\ell_i^{\text{cls}} = (1-\alpha)\ell_i^{\text{ce}} + \alpha\ell_i^{\text{soft}}. \tag{5}$$

We sweep the kernel width $w \in \{1, 5, 10, 20\}$ and the soft-target mixing weight $\alpha \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$. Based on these sweeps, we use a Gaussian kernel with $w = 5$ and $\alpha = 0.9$ in the final configuration.

B.2 Expected-Coordinate Regression

As an alternative way to impose metric structure on the coordinate vocabulary, we compute a differentiable coordinate estimate from the location-token distribution,

$$\hat{v}_i = \sum_{j=0}^{K-1} q_{i,j} \frac{j}{K-1}. \tag{6}$$

We then regress $\hat{v}_i$ to the target coordinate v_i using one of two number-token loss (NTL) variants:

$$\ell_i^{\text{ntl}} = \begin{cases} |\hat{v}_i - v_i|, & \ell_1, \\ (\hat{v}_i - v_i)^2, & \text{MSE}. \end{cases} \tag{7}$$

These objectives shape the probability distribution over coordinate bins without introducing an additional regression head. For each NTL variant, we sweep its auxiliary loss weight over

$$\lambda_{\text{ntl}} \in \{1.0,\ 0.1,\ 10^{-3},\ 10^{-4},\ 10^{-5}\}. \tag{8}$$

MSE performs worse than ℓ_1, so the main paper reports only the ℓ_1 variant.

B.3 Box-Level Generalized IoU

We additionally evaluate a box-level Generalized IoU (GIoU) objective following Rezatofighi et al. (2019). Given the predicted box $\hat{B}$ and target box B, GIoU augments IoU with a penalty determined by the smallest enclosing box C,

$$\text{GIoU}(\hat{B}, B) = \text{IoU}(\hat{B}, B) - \frac{|C| - |\hat{B} \cup B|}{|C|}, \qquad \ell^{\text{giou}} = 1 - \text{GIoU}(\hat{B}, B). \tag{9}$$

This directly couples all four predicted coordinates and provides a geometric training signal even when the predicted and target boxes do not overlap. Predicted coordinates are obtained through the differentiable expectation over the location-token distribution. Since these soft coordinate estimates can produce inverted boxes early in training, we additionally apply an ordering penalty,

$$\ell^{\text{ord}} = [\hat{x}_1 - \hat{x}_2]_+ + [\hat{y}_1 - \hat{y}_2]_+, \tag{10}$$

while retaining the semantic ordering of the four coordinate tokens. We sweep the GIoU loss weight over $\{1.0,\ 0.1,\ 10^{-3},\ 10^{-4},\ 10^{-5}\}$ and select $\lambda_{\text{giou}} = 10^{-3}$ based on the validation sweep.

Final configuration. The ablations in the main paper compare these geometry-aware objectives under the same training setup. Both in ablations and at scale, Gaussian distance-aware soft cross-entropy provides the strongest overall grounding performance, and we therefore retain it as the sole auxiliary grounding objective in the final FastVLM2 training recipe.

C Training-Free Confidence Estimation

This section provides the complete formulation and additional ablations for the training-free confidence estimator used in Section 4.3. The method operates only on the location-token logits already produced during decoding; it introduces no additional parameters, requires no additional forward pass, and does not modify the predicted boxes. FastVLM2-1.8B and Rex-Omni are evaluated with their SFT+RL checkpoints, and LocateAnything-3B with its SFT-only checkpoint.

C.1 FORMULATION

Let a predicted box be $\hat{B} = (\hat{x}_1, \hat{y}_1, \hat{x}_2, \hat{y}_2)$, represented by four discrete location tokens from a vocabulary of $K = 1000$ coordinate bins. At decode position $t \in \{1, 2, 3, 4\}$, we restrict the model logits to the location-token vocabulary and compute

$$p_t(k) = \frac{\exp(z_{t,s+k})}{\sum_{j=0}^{K-1} \exp(z_{t,s+j})}, \qquad k = 0, \ldots, K-1, \tag{11}$$

where s is the vocabulary index of the first location token. Let $\hat{c}_t$ denote the emitted coordinate bin. A per-coordinate confidence C_t is computed from p_t, and the four values are combined using the arithmetic mean,

$$S_B = \frac{1}{4} \sum_{t=1}^{4} C_t. \tag{12}$$

Thus, all confidence scores are obtained directly from distributions that are already available during autoregressive decoding.

Bin-space scores. We first consider scores that operate directly on the coordinate-token distribution. The emitted-token probability and top-two probability margin are

$$C_t^{\text{tok}} = p_t(\hat{c}_t), \tag{13}$$

$$C_t^{\text{mar}} = p_t^{(1)} - p_t^{(2)}, \tag{14}$$

where $p_t^{(1)}$ and $p_t^{(2)}$ are the two largest probabilities. We also evaluate normalized entropy,

$$C_t^{\text{ent}} = 1 - \frac{H_t}{\log K}, \qquad H_t = -\sum_{k=0}^{K-1} p_t(k) \log p_t(k), \tag{15}$$

and two scores that use the ordered structure of the coordinate vocabulary:

$$C_t^{\text{dist}} = 1 - \frac{1}{K-1} \sum_{k=0}^{K-1} p_t(k) |k - \hat{c}_t|, \tag{16}$$

$$C_t^{\text{ker}} = \sum_{k=0}^{K-1} p_t(k) \exp\left(-\frac{(k - \hat{c}_t)^2}{2\sigma^2} \right). \tag{17}$$

We use $\sigma = 5$ bins for the Gaussian kernel. These scores measure uncertainty in coordinate-bin space, but do not account for the scale or geometry of the predicted box.

Box-space scores. To incorporate box geometry, let $B_{t \leftarrow k}$ denote the predicted box obtained by replacing coordinate t with candidate bin k while holding the remaining three coordinates fixed. We consider the expected IoU

$$C_t^{\text{IoU}} = \sum_{k=0}^{K-1} p_t(k) \ \text{IoU}\left(B_{t \leftarrow k}, \hat{B} \right), \tag{18}$$

and the probability mass assigned to coordinate alternatives that preserve the predicted box above an IoU threshold τ,

$$C_t^{\tau} = \sum_{k=0}^{K-1} p_t(k) \, \mathbf{1}\left[\text{IoU}\left(B_{t \leftarrow k}, \hat{B} \right) \geq \tau \right]. \tag{19}$$

Candidates producing invalid boxes, i.e., $x_2 \leq x_1$ or $y_2 \leq y_1$, are assigned zero IoU. Unlike bin-space scores, Eqs. (18) and (19) are scale-aware: the same coordinate perturbation has a larger effect on the IoU of a small box than on that of a large box.

Baselines. We compare against three score-free or nearly score-free baselines. *Random* assigns an independent random score to each box, *constant* assigns every box the same confidence and therefore reproduces score-free evaluation, and *emission order* ranks boxes according to autoregressive generation order, $S = 1/(1 + \text{rank})$. The same predicted boxes are used for all ranking-only rows below; deduplication and confidence-threshold rows are reported separately because they alter the retained detection set.

Score	Family	mAP	AP_{50}	AP_S	AP_M	AP_L	$F1_{50}$	$F1_{95}$	F1 Avg.	P_{50}	R_{50}
Baselines											
Random	baseline	35.26	55.12	16.15	41.08	60.08	71.61	15.30	52.78	74.71	68.76
Constant	baseline	36.05	56.17	16.74	42.19	60.28	71.54	15.05	52.76	74.55	68.76
Emission order	baseline	38.08	58.69	16.95	43.86	62.84	72.48	15.89	53.63	76.63	68.76
Bin-space, order-blind											
C^{tok}	bin-space	39.88	59.65	18.39	46.71	64.31	72.50	16.67	54.28	76.68	68.76
C^{mar}	bin-space	37.03	56.70	16.78	43.52	61.27	71.63	15.26	52.84	74.77	68.75
C^{ent}	bin-space	41.20	61.12	19.07	47.80	66.11	72.71	16.89	54.46	77.14	68.76
Bin-space, distance-aware											
C^{dist}	bin-space	40.97	61.23	19.05	47.54	66.91	72.09	15.98	53.51	75.77	68.75
C^{ker}	bin-space	40.76	60.99	19.13	47.69	66.06	72.69	16.28	54.27	77.11	68.76
Box-space											
C^{IoU}	box-space	44.51	63.37	21.20	51.73	70.60	73.09	19.10	55.53	78.02	68.76
$C^{\tau=0.50}$	box-space	43.53	63.52	20.21	48.83	68.98	72.80	17.36	54.43	77.35	68.75
$C^{\tau=0.75}$	box-space	44.22	63.68	20.91	49.86	69.85	73.17	17.52	55.08	78.19	68.75
$C^{\tau=0.90}$	box-space	44.57	63.27	21.37	51.53	70.41	73.14	18.91	55.89	78.13	68.75
$C^{\tau=0.95}$	box-space	44.35	62.87	21.31	51.93	70.38	72.83	19.42	55.95	77.42	68.76
Deduplication of $C^{\tau=0.90}$											
+ NMS	dedup	44.89	63.77	21.39	51.36	71.02	73.63	18.98	56.24	79.58	68.50
Confidence threshold θ *applied to* $C^{\tau=0.90}$											
$\theta = 0.1$	threshold	44.57	63.27	21.37	51.53	70.41	73.14	18.91	55.89	78.13	68.75
$\theta = 0.2$	threshold	44.56	63.25	21.36	51.53	70.41	73.22	18.91	55.90	78.37	68.71
$\theta = 0.3$	threshold	44.53	63.12	21.32	51.53	70.41	73.34	18.91	55.94	78.89	68.52
$\theta = 0.4$	threshold	44.41	62.74	21.10	51.53	70.41	73.40	18.91	56.00	79.76	67.98
$\theta = 0.5$	threshold	44.11	61.76	20.57	51.49	70.40	73.22	18.90	56.07	81.00	66.80
$\theta = 0.6$	threshold	43.46	60.08	19.09	51.42	70.38	72.53	18.90	55.99	82.59	64.65
$\theta = 0.7$	threshold	42.38	57.52	16.81	51.02	70.32	71.34	18.89	55.79	85.01	61.45

Table 6: **Training-free confidence ablations for FastVLM2-1.8B on COCO `val2017`.** Ranking-only rows score an identical set of predicted boxes. Deduplication and thresholding additionally modify the retained detection set.

C.2 FastVLM2 Confidence Ablations

Table 6 reports the complete confidence ablation for the FastVLM2 checkpoint used in the main paper. Values are multiplied by 100. Box-space scoring provides the strongest ranking-only results: $C^{\tau=0.90}$ reaches 44.57 mAP and expected-IoU weighting reaches 44.51, compared with 36.05 for score-free ranking and 38.08 for emission order. Thus, $C^{\tau=0.90}$ improves mAP by 8.52 points over the score-free protocol without changing the predicted boxes. Applying NMS using the training-free confidence scores further increases mAP to 44.89, although this row is not a pure ranking comparison because detections are removed.

Bin-space versus box-space. The best bin-space score, C^{ent}, reaches 41.20 mAP, whereas the best ranking-only box-space score reaches 44.57, a 3.37 AP gap. The box-space gain is also consistent across object scales. Relative to random ranking, $C^{\tau=0.90}$ improves AP_S, AP_M, and AP_L by approximately $1.32\times$, $1.25\times$, and $1.17\times$, respectively. This behavior is consistent with the geometry- and scale-aware construction of Eq. (19), which aggregates probability mass according to its effect on the resulting box rather than treating coordinate-bin uncertainty independently.

IoU threshold τ. The thresholded box-space score is robust across a broad range of τ. The strongest ranking result is obtained at $\tau = 0.90$ with 44.57 mAP, while $\tau = 0.95$ remains close at 44.35 mAP and yields the strongest $F1_{95}$ and F1 Avg. among the ranking-only box-space variants. The continuous expected-IoU score similarly reaches 44.51 mAP, indicating that the benefit of box-space aggregation is not specific to a single IoU threshold.

Deduplication and confidence thresholds. NMS increases mAP from 44.57 to 44.89. Confidence thresholding exposes the expected precision-recall trade-off: increasing θ raises P_{50} from 78.13 at $\theta = 0.1$ to 85.01 at $\theta = 0.7$, while R_{50} decreases from 68.75 to 61.45. $F1_{50}$ peaks at 73.40 for $\theta = 0.4$, while F1 Avg. peaks at 56.07 for $\theta = 0.5$.

Coordinates	$\lvert\cdot\rvert$	mAP	AP_{50}	AP_S	AP_M	AP_L	$F1_{50}$	P_{50}	R_{50}
x_1	1	30.55	50.10	8.89	31.48	55.12	61.52	65.32	58.14
y_1	1	31.43	50.70	9.97	33.15	56.38	61.86	66.11	58.13
x_2	1	30.89	50.09	9.16	31.82	55.97	61.72	65.78	58.13
y_2	1	31.41	50.60	9.99	33.65	56.00	61.64	65.60	58.13
x_1, x_2	2	31.45	50.79	9.23	32.59	56.81	61.96	66.33	58.13
y_1, y_2	2	32.05	51.21	10.08	34.00	57.85	61.97	66.36	58.13
x_1, y_1	2	32.00	51.41	10.12	34.28	57.43	62.10	66.66	58.13
x_2, y_2	2	32.15	51.24	10.44	34.55	58.05	62.14	66.73	58.14
all but y_2	3	32.44	51.67	10.30	34.55	58.67	62.28	67.07	58.13
all but x_1	3	32.51	51.57	10.53	34.83	58.70	62.16	66.80	58.13
all four	4	32.77	51.89	10.50	35.19	59.36	62.28	67.07	58.13

Table 7: **Coordinate-subset ablation for box-space confidence.** Results use $C^{\tau=0.90}$ on the diagnostic checkpoint from the initial confidence study, not the checkpoint used in the main paper. The full four-coordinate score gives the strongest overall mAP.

C.3 Additional Coordinate and Reduction Ablations

The following ablations were performed on an earlier diagnostic checkpoint from the initial confidence study. We retain them because they isolate properties of the confidence estimator, while Table 6 provides the absolute performance numbers for the stronger FastVLM2 checkpoint used in the main paper.

Reduction across coordinates. On the diagnostic checkpoint, arithmetic and geometric averaging give essentially identical mAP (32.77 for both), whereas taking the minimum coordinate confidence reduces mAP to 32.42. We therefore use the arithmetic mean in Eq. (12) for simplicity.

Coordinate subsets. Table 7 evaluates the contribution of individual coordinates using $C^{\tau=0.90}$. Performance increases as additional coordinates are incorporated, and the full four-coordinate score gives the strongest overall mAP. This indicates that the gain is not explained by a single privileged coordinate.

C.4 Transfer to Rex-Omni

To test whether the estimator depends on FastVLM2-specific training, we apply the same scoring functions without modification to Rex-Omni. Rex-Omni uses the same single-token coordinate representation with approximately 1000 bins, allowing the confidence computation to transfer directly. Table 8 reports the complete COCO `val2017` results.

The transfer result is consistent with FastVLM2. The best box-space score reaches 44.20 mAP, improving over constant scoring by 8.56 AP and over emission order by 7.07 AP. It also increases $F1_{95}$ from 15.84 to 19.67, a relative improvement of approximately 24%. In contrast to FastVLM2, deduplication does not improve mAP, indicating that the usefulness of NMS depends on the detector's duplicate-prediction rate rather than on the confidence formulation itself.

C.5 Transfer to LocateAnything

We further evaluate LocateAnything-3B, which uses parallel box decoding rather than autoregressive coordinate decoding. All four coordinates are predicted in a single forward pass, making the per-coordinate marginal sweep in Eqs. (18) and (19) exact under its decoder. The confidence estimator and evaluation harness are otherwise unchanged.

The same box-space estimator again improves confidence ranking. The best box-space result reaches 42.90 mAP, a gain of 8.34 AP over score-free ranking and 6.71 AP over emission order. The best bin-space result reaches 40.47 mAP, narrowing the bin-space versus box-space gap to 2.43 AP. Tight-localization performance also improves substantially: $F1_{95}$ increases from 18.93 for constant scoring to 22.45 with $C^{\tau=0.95}$.

Score	mAP	AP_{50}	AP_S	AP_M	AP_L	$F1_{50}$	$F1_{95}$	R_{50}
Baselines								
Random	35.29	55.72	16.69	39.81	59.64	72.13	15.72	67.90
Constant	35.64	56.30	17.05	40.64	60.09	72.01	15.84	67.90
Emission order	37.13	57.87	17.12	42.11	61.55	72.30	16.41	67.90
Bin-space, order-blind								
C^{tok}	39.24	59.47	18.42	45.22	64.41	72.38	17.03	67.90
C^{mar}	37.16	57.52	17.27	42.35	61.41	72.19	15.77	67.91
C^{ent}	39.44	59.71	18.52	45.33	65.71	72.47	17.09	67.90
Bin-space, distance-aware								
C^{rdist}	39.56	60.20	18.64	45.52	66.71	72.40	16.83	67.90
C^{ker}	38.93	59.44	18.53	45.35	65.27	72.41	16.87	67.90
Box-space								
C^{IoU}	44.20	62.83	20.82	49.57	70.10	72.70	19.67	67.90
$C^{\tau=0.50}$	42.44	62.73	19.49	46.55	68.53	72.53	17.81	67.90
$C^{\tau=0.75}$	42.95	62.74	19.93	47.12	68.86	72.71	17.40	67.90
$C^{\tau=0.90}$	43.73	62.69	20.79	48.34	69.03	72.89	18.35	67.90
$C^{\tau=0.95}$	44.01	62.54	20.96	49.31	69.31	72.84	19.65	67.90
Deduplication of $C^{\tau=0.90}$								
+ NMS	43.63	62.53	20.77	48.18	68.88	72.83	18.33	67.70
Confidence threshold θ *on* $C^{\tau=0.90}$								
$\theta = 0.5$	43.71	62.56	20.75	48.33	69.03	73.06	18.35	67.74
$\theta = 0.7$	43.42	61.75	20.44	47.99	68.95	73.28	18.36	66.61

Table 8: **Training-free confidence estimation on Rex-Omni.** The identical scoring functions are applied without retraining or model-specific fitting.

Score	mAP	AP_{50}	AP_S	AP_M	AP_L	$F1_{50}$	$F1_{95}$	R_{50}
Baselines								
Random	34.24	51.70	17.32	38.81	55.73	68.92	18.72	63.31
Constant	34.56	52.09	17.51	39.09	55.93	68.79	18.93	63.33
Emission order	36.19	53.92	17.80	40.78	57.95	69.16	19.45	63.33
Bin-space, order-blind								
C^{tok}	39.54	56.89	20.37	44.68	61.90	69.52	20.10	63.32
C^{mar}	36.95	54.27	18.81	41.93	57.68	69.14	18.94	63.32
C^{ent}	39.55	56.84	20.07	44.88	62.71	69.39	19.83	63.33
Bin-space, distance-aware								
C^{rdist}	39.60	57.00	19.54	44.33	64.08	69.51	19.53	63.31
C^{ker}	40.47	57.60	20.45	45.72	64.59	69.93	20.41	63.32
Box-space								
C^{IoU}	42.81	59.20	21.52	47.05	66.37	70.30	21.42	63.33
$C^{\tau=0.50}$	41.10	58.57	20.52	45.08	63.63	70.11	20.63	63.33
$C^{\tau=0.75}$	42.18	59.13	21.21	46.25	65.19	70.39	20.83	63.33
$C^{\tau=0.90}$	42.90	59.15	22.05	47.13	66.17	70.30	21.94	63.33
$C^{\tau=0.95}$	42.87	58.85	22.30	47.61	66.36	70.20	22.45	63.33
Confidence threshold θ *on* $C^{\tau=0.95}$								
$\theta = 0.2$	42.66	58.19	21.88	47.59	66.36	70.52	22.45	62.42
$\theta = 0.5$	39.57	51.21	16.10	45.68	66.08	66.64	22.59	54.05
$\theta = 0.7$	33.88	41.61	8.06	38.60	63.22	58.51	22.74	43.23

Table 9: **Training-free confidence estimation on LocateAnything-3B.** The full set of confidence and operating-point results is shown.

C.6 Cross-Model Summary

Table 3 summarizes the ranking-only results across three independently trained grounding models. Box-space confidence consistently provides the strongest ranking performance, improving mAP over the score-free protocol by 8.34–8.56 points without retraining or model-specific fitting. In contrast, the gains from the best bin-space score vary more substantially across models. Box-space scoring also improves $F1_{50}$ by 0.88–1.60 points over the score-free baseline. Overall, the similar mAP gains across FastVLM2-1.8B, Rex-Omni-3B, and LocateAnything-3B indicate that the proposed confidence estimator transfers reliably across independently trained models that use quantized location tokens.

Task	Output	Question Template
Dense Detection	Multiple Boxes	Detect all the objects in the image that belong to the category set: {categories}. Detect the following categories in the image: {categories}. Locate every instance of these categories: {categories}. Find all of {categories} in the image and give their bounding boxes.
Phrase Grounding / Referring Expression	Single Box	Locate a single instance that matches the following description: {q}. Please provide the bounding box coordinate of the region this sentence describes: {q}. Give the bounding box coordinates for the region described by: {q}. Where is {q}? Output its bounding box. Detect the region described as: {q}.
Pointing	Point	Point to: {q}.

Table 10: Grounding prompt templates used during training. We sample from multiple instruction templates for each grounding task. {categories} denotes the queried set of object categories, while {q} denotes a free-form referring expression or grounding query.

D Training Details

D.1 Training Datasets

Our training data mixture consists of PixMo-Cap, PixMo-Docs, PixMo-Points, and PixMo-Count (Deitke et al., 2025), LLaVA-ReCap-CC3M and LLaVA-ReCap-CC12M (Li et al., 2024a), CoSyn-400k (Yang et al., 2025b), Sujet-Finance-QA-Vision-100k (Sujet AI, 2024), Leopard-Instruct (Jia et al., 2025), LLaVA-1.5 (Liu et al., 2024a), InternVL (Chen et al., 2024), LLaVA-OneVision (Li et al., 2025a), Eagle-1.8M (Shi et al., 2025), LNQA (Pont-Tuset et al., 2020), EMOVA-SFT-4M (Chen et al., 2025), Cambrian-1 (Tong et al., 2024), and the single-image subset of MAmmoTH-VL-Instruct-12M (Guo et al., 2025). For grounding, we additionally include the training splits of RefCOCO, RefCOCO+ (Kazemzadeh et al., 2014), RefCOCOg (Mao et al., 2016), and gRefCOCO (Liu et al., 2023a), the single-image grounding data from LLaVA-ST (Li et al., 2025b), Toloka (Ustalov et al., 2023), and ABACUS (Mondal et al., 2026), and the COCO (Lin et al., 2014), OpenImages v6 (Kuznetsova et al., 2020), V3Det (Wang et al., 2023), Objects365 (Shao et al., 2019), RoboAfford (Tang et al., 2025), D3 (Xie et al., 2023), RexThinker-HumanRef (Jiang et al., 2025a), and PartImageNet++ (Li et al., 2024b) subsets shipped as part of LocateAnything (Wang et al., 2026). We use only training splits and exclude data sources based on their documented dataset provenance.

Negative sampling, box ordering, and prompting. Following the data-construction protocol of LocateAnything (Wang et al., 2026), we augment the OpenImages and COCO grounding data with negative categories sampled from classes that are not annotated in the corresponding image. Approximately 30% of annotation records contain *partial negatives*, in which negative categories are mixed with valid target categories, while an additional 15% contain *strict negatives*, for which none of the queried categories correspond to ground-truth annotations and the expected response is None for every listed category. This exposes the model to queries with partially or entirely absent targets and encourages appropriate abstention. We additionally follow the X-Y Corner ordering of LocateAnything (Wang et al., 2026): when multiple boxes are associated with a query, boxes are sorted by the x-coordinate of the top-left corner and then by its y-coordinate, providing a consistent serialization order during training. For each grounding task, we sample from multiple task-specific instruction templates; the templates used for dense detection, phrase grounding/referring expressions, and pointing are summarized in Table 10.

D.2 Training Protocol for Ablations

Unless otherwise specified, all ablation experiments use the three-stage training setup summarized in Table 11. All models are trained on multiple nodes of NVIDIA H100-80GB GPUs. For the native-resolution ablations, we first study the image-processing strategy across training stages and determine the final recipe in Table 1b. Based on these experiments, all subsequent runs use fixed-resolution image preprocessing during Stage 1 and Stage 1.5, followed by native-resolution training during Stage 2.0. This preprocessing recipe is held fixed for the remaining experiments so that individual architectural and training interventions can be evaluated under a common setup. All ab-

	Stage 1.0	Stage 1.5	Stage 2.0
Learning rate	1×10^{-3}	4×10^{-5}	4×10^{-5}
Batch size	128	512	512
LR schedule	cosine decay	cosine decay	cosine decay
LR warmup ratio	0.03	0.03	0.03
Optimizer	AdamW	AdamW	AdamW
Max. Context Length	2k	2k	8k
Steps	4k	27k	15k
Trainable modules	Projector	Full Model	Full Model

Table 11: **Three-stage training protocol used for ablation studies.** We report model performance after Stage 2.0, the instruction-tuning stage.

lations in Sections 4.1 and 4.2 use the optimization hyperparameters and training schedule reported in Table 11.

D.3 SFT Training Details

For the best-performing models reported in Table 4, we extend the training schedule used in ablations with two additional instruction-tuning stages, Stage 3.0 and Stage 4.0. We find that repeated exposure to grounding data is beneficial for detection and dense grounding performance. Accordingly, in these final two SFT stages we reuse all grounding datasets from the earlier stages and skew the training distribution toward dense grounding tasks. The full training schedule is detailed in Table 12. The checkpoint obtained after Stage 4.0 is reported as **FastVLM2-SFT** in Appx. F. We then further fine-tune this model with reinforcement learning, as described in Appx. D.5.

	Stage 1.0	Stage 1.5	Stage 2.0	Stage 3.0	Stage 4.0
Learning rate	1×10^{-3}	4×10^{-5}	4×10^{-5}	4×10^{-5}	2×10^{-5}
Batch size	128	512	512	512	512
LR schedule	cosine decay	cosine decay	cosine decay	cosine decay	cosine decay
LR warmup ratio	0.03	0.03	0.03	0.03	0.03
Optimizer	AdamW	AdamW	AdamW	AdamW	AdamW
Max. Context Length	2k	2k	8k	8k	8k
Steps	5k	38k	33k	33k	13k
Image preprocessing	Fixed res.	Fixed res.	Native res.	Native res.	Native res.
Grounding aux. loss	–	–	✓	✓	✓
Trainable modules	Projector	Full Model	Full Model	Full Model	Full Model

Table 12: **Training protocol for FastVLM2.** We use five training stages for the best-performing models reported in Table 4. Results after Stage 4.0 are reported as FastVLM2-SFT in Appx. F.

D.4 Native-Resolution Training

Native-resolution inputs have different spatial extents and therefore cannot be directly stacked into a conventional dense batch. In our experiments, we simply process images individually through the vision encoder before batching the resulting visual tokens for the language model. This eager implementation supports arbitrary input shapes and is sufficient for the results reported in this paper. However, frameworks that rely on JIT-compiled or static computation graphs can incur recompilation overhead when the input shape changes across examples.

One possible alternative is to embed variable-resolution images into a fixed-size *canvas* and propagate a validity mask through the encoder. The mask can be used to exclude padded regions from spatially aggregating operations, including convolutions, normalization, squeeze-excitation, and self-attention, while invalid output tokens are discarded before being passed to the language model. Mask-aware variants of these operations are well established, so this does not require architectural changes specific to our model. The canvas trades additional computation on padded regions for a fixed input shape, enabling reuse of a compiled training graph; padding overhead can further be reduced by bucketing examples into a small number of canvas shapes or aspect ratios. Such a

Benchmark	`lmms-eval` task	Reported metric key
General image understanding		
AI2D	`ai2d`	`exact_match,flexible-extract`
SEED-Bench	`seedbench`	`seed_image,none`
RealWorldQA	`realworldqa`	`exact_match,flexible-extract`
Text-rich / document		
TextVQA (val)	`textvqa`	`exact_match,none`
ChartQA	`chartqa`	`relaxed_overall,none`
DocVQA (val)	`docvqa`	`anls,none`
InfoVQA (val)	`infovqa`	`anls,none`
OCRBench	`ocrbench`	`ocrbench_accuracy,none`
Referring expression (reported as RefCOCO Avg., the arithmetic mean of the 8 splits)		
RefCOCO val / testA / testB	`refcoco_bbox_rec_{val,testA,testB}`	`refcoco_ACC@0.5,none`
RefCOCO+ val / testA / testB	`refcoco+_bbox_rec_{val,testA,testB}`	`refcoco_ACC@0.5,none`
RefCOCOg val / test	`refcocog_bbox_rec_{val,test}`	`refcoco_ACC@0.5,none`
Referring expression, Rex-Omni protocol (outside `lmms-eval`)		
RefCOCOg val / test	—	F1 @ IoU 0.5, F1 @ mIoU
HumanRef	—	F1 @ IoU 0.5, F1 @ mIoU
Dense prediction, Rex-Omni protocol (outside `lmms-eval`)		
COCO (val2017)	—	F1 @ IoU 0.5, F1 @ IoU 0.95, F1 @ IoU Mean
LVIS (v1 val)	—	F1 @ IoU 0.5, F1 @ IoU 0.95, F1 @ IoU Mean

Table 13: Benchmarks, `lmms-eval` v0.7.1 task names, and the exact metric key reported for each.

formulation can be implemented in commonly used training frameworks such as PyTorch, JAX, or TensorFlow.

D.5 Reinforcement Learning Training

After SFT, we further optimize FastVLM2 with reinforcement learning following the Group Sequence Policy Optimization (GSPO) (Zheng et al., 2025) recipe of Vero (Sarch et al., 2026). We train on a mixture of grounding data (COCO (Lin et al., 2014), RefCOCO (Kazemzadeh et al., 2014)) and general instruction-following data (the Vero-600k mixture (Sarch et al., 2026) with same filtering criteria mentioned in Appx. D.1). For grounding, similar to Rex-Omni (Jiang et al., 2026), the reward is the F_1 score averaged over IoU thresholds from 0.5 to 0.95 in steps of 0.05, with boxes matched greedily and one-to-one within each category. For general tasks, we use Qwen3-32B (Yang et al., 2025a) as judge rather than Vero's own rule-based grader, which is brittle to the format inconsistencies of an SFT checkpoint's answers. Verifiable tasks (numeric, multiple-choice, string/list matching, counting, web action) receive a strict binary-equivalence rubric, while open-ended and instruction-following tasks share a graded quality rubric adapted from Vero (Sarch et al., 2026).

Unlike Vero, we request no reasoning scaffold: every rollout uses the single-line system prompt `You are a helpful assistant.`, since a scaffold-requesting prompt reduced the fraction of rollouts receiving non-zero reward and increased response length. We otherwise use Vero's hyperparameters (Sarch et al., 2026): 8 rollouts per prompt, sampling temperature 1.0, and learning rate 10^{-6}. We depart from Vero in three ways: no learning-rate warmup, no format or overlong-length reward, and shorter prompt/response limits of $8{,}192$ and $1{,}024$ tokens (Vero uses $12{,}288$ and $16{,}384$). We train for 500 steps on 8 H100 GPUs with all parameters trainable.

E Evaluation Details

We compare FastVLM2 against three categories of methods: (1) *generalist vision-language models*, including InternVL3.5 (Wang et al., 2025), MiniCPM-V (Yu et al., 2025), Qwen3-VL (Bai et al., 2025a), Ovis2.5 (Lu et al., 2025), and Molmo2 (Clark et al., 2026), which are trained for broad instruction-following and evaluated on both standard VLM benchmarks and grounding; (2) *grounding-specialist vision-language models*, including Rex-Omni (Jiang et al., 2026) and LocateAnything (Wang et al., 2026), which are trained and evaluated specifically on grounding tasks; and (3) *specialized object detectors*, including Grounding DINO (Liu et al., 2023b), DETR (Carion et al., 2020), Deformable-DETR (Zhu et al., 2021), and DINO (Zhang et al., 2023), which are included in Tables 5 and 15 as closed- and open-set reference points rather than direct competitors.

Models	Vision Encoder		ChartQA	OCRBench	TextVQA	DocVQA	InfoVQA	AI2D	RealWorldQA	SEED-Bench	RefCOCO-Avg
	Name	Size (M)									
InternVL3.5-1B	InternViT	304	67.4	**74.4**	**68.3**	80.8	57.6	65.0	59.6	**72.7**	70.4
FastVLM2-0.7B - SFT	FastViTHD	**125**	77.8	64.4	63.3	91.9	**72.5**	77.1	60.0	69.6	89.8
FastVLM2-0.7B	FastViTHD	**125**	**79.3**	67.0	65.7	**92.4**	71.7	**78.1**	**62.0**	70.9	**89.9**
MiniCPM-V-4.6-1.3B	SigLIP2	418	64.4	77.7	73.4	88.0	62.9	77.2	61.8	76.5	70.3
InternVL3.5-2B	InternViT	304	78.6	79.5	74.6	85.6	65.7	75.2	61.6	75.7	82.4
Qwen3-VL-2B	QwenViT	306	79.9	81.7	79.6	91.2	69.7	75.8	64.2	76.1	82.1
Ovis2.5-2B	SigLIP2	414	**85.7**	**85.5**	**80.0**	94.0	76.5	83.3	**67.8**	**77.0**	83.2
FastVLM2-1.8B - SFT	FastViTHD	**125**	81.0	71.6	67.0	96.2	**84.5**	**84.4**	62.2	71.3	90.8
FastVLM2-1.8B	FastViTHD	**125**	82.2	72.4	68.8	**96.3**	83.6	83.4	63.5	72.8	**91.0**
InternVL3.5-4B	InternViT	304	83.0	79.5	75.1	89.1	74.6	79.8	65.8	76.5	86.8
Molmo2-4B	SigLIP2	383	67.5	70.1	76.7	83.5	66.9	83.9	**74.5**	76.8	–
Qwen3-VL-4B (instruct)	QwenViT	306	83.8	**85.4**	**79.6**	94.7	78.7	84.1	71.5	**78.8**	89.0
FastVLM2-4B - SFT	FastViTHD	**125**	83.4	78.4	70.3	**98.0**	92.4	**86.0**	63.5	74.1	92.4
FastVLM2-4B	FastViTHD	**125**	**84.1**	80.2	72.6	97.9	**93.3**	85.7	64.7	74.9	**92.7**

Table 14: Per-benchmark VQA accuracy and RefCOCO performance underlying the averages reported in Table 4. RefCOCO-Avg is the arithmetic mean of ACC@0.5 across the eight RefCOCO / RefCOCO+ / RefCOCOg splits (Appx. E); this is distinct from the F1-based Avg. Refer. Exp. column of Table 4, which is computed only from RefCOCOg and HumanRef (Table 15). Within each model group, **bold** indicates the best result and underlined indicates the second-best result among reported values.

All general vision-language numbers are produced with `lmms-eval` v0.7.1 under a single shared harness, so every model in the paper sees identical prompts, decoding settings, and answer parsers. Table 13 lists each benchmark, covering general image understanding (AI2D (Kembhavi et al., 2016), SEED-Bench (Li et al., 2023), and RealWorldQA) and text-rich document understanding (TextVQA (Singh et al., 2019), ChartQA (Masry et al., 2022), DocVQA (Mathew et al., 2021), InfoVQA (Mathew et al., 2022), and OCRBench (Liu et al., 2024c)), together with the exact `lmms-eval` task name and the metric key we read out of that task's result file; we report exactly one scalar per benchmark and never rescore or re-parse model outputs afterwards. For referring-expression grounding, we run the eight standard RefCOCO / RefCOCO+ (Kazemzadeh et al., 2014) / RefCOCOg (Mao et al., 2016) splits and report RefCOCO Avg., the arithmetic mean of the eight ACC@0.5 values.

For dense grounding, we report F1 on COCO (Lin et al., 2014) and LVIS (Gupta et al., 2019) using standard IoU-based matching between predicted and ground-truth boxes. For FastVLM2, Rex-Omni, and LocateAnything, which represent each box using discrete location tokens, we additionally report mAP using our training-free confidence estimator. Specifically, we compute a per-box confidence from the model's coordinate-token distributions using C^τ with $\tau = 0.9$ (Eq. (1)), which measures the probability mass assigned to coordinate alternatives that preserve the predicted box at $\mathrm{IoU} \geq 0.9$, and apply class-wise NMS to the resulting scored boxes. This procedure requires no additional parameters or forward passes. In contrast, models such as Qwen3-VL and Ovis2.5 generate coordinates as plain text, for which there is no single distribution over ordered coordinate bins; our confidence estimator is therefore not directly applicable to these models, and we do not report mAP for them under this protocol. For LocateAnything, we were unable to reproduce the paper's best reported configuration due to compilation issues with the `MagiAttention` dependency; consequently, the performance from our re-run differs from the numbers reported in the original work.

RefCOCOg val, RefCOCOg test (Mao et al., 2016), and HumanRef (Jiang et al., 2025b) are additionally evaluated outside `lmms-eval` with Rex-Omni's evaluation library, which is the protocol those benchmarks are reported under in the grounding literature. There we report **F1 @ IoU 0.5** and **F1 @ mIoU**, the latter being the library's mIoU column: F1 averaged over the IoU thresholds $0.50:0.05:0.95$.

F Detailed Main Results

This section reports the per-benchmark numbers underlying the averages in Table 4. Table 14 gives per-benchmark VQA accuracy and RefCOCO performance, underlying the Avg. VQA Perf. column, and Table 15 gives per-benchmark referring-expression results on RefCOCOg and HumanRef, un-

Method	Score Thresh.	RefCOCOg val		RefCOCOg test		HumanRef	
		F1@IoU 0.5	F1@mIoU	F1@IoU 0.5	F1@mIoU	F1@IoU 0.5	F1@mIoU
Open-set Specialized Detector							
Grounding DINO-Swin-T	0.25	52.9	45.9	53.8	46.8	28.0	25.2
Vision-Language Models							
DeepSeek-VL2-Tiny	–	67.4	50.5	69.3	52.1	39.1	31.4
InternVL3.5-1B	–	82.7	62.4	82.3	62.7	50.1	29.9
InternVL3.5-2B	–	86.0	67.1	85.7	67.7	44.5	21.4
InternVL3.5-4B	–	89.5	75.3	89.3	75.9	57.5	38.1
Ovis2.5-2B	–	87.4	73.4	87.6	73.8	70.6	50.0
Qwen3-VL-2B	–	84.5	71.3	85.4	72.2	70.6	63.5
Qwen3-VL-4B	–	88.0	74.7	87.6	74.6	77.7	71.1
Rex-Omni-3B (Grounding Specialist)	–	86.6	73.6	86.8	74.3	**85.4**	**79.9**
LocateAnything-3B (Grounding Specialist)	–	88.6	76.7	88.8	77.6	<u>82.9</u>	<u>78.7</u>
FastVLM2-0.7B - SFT	–	89.7	77.9	89.6	78.7	76.3	64.5
FastVLM2-1.8B - SFT	–	90.6	80.6	90.6	81.6	77.9	68.5
FastVLM2-4B - SFT	–	<u>91.1</u>	<u>81.9</u>	<u>91.4</u>	<u>83.1</u>	79.8	68.4
FastVLM2-0.7B	–	90.1	79.7	89.8	80.3	79.0	71.0
FastVLM2-1.8B	–	90.4	81.2	91.0	82.5	79.4	72.1
FastVLM2-4B	–	**91.5**	**84.3**	**92.4**	**85.8**	82.1	74.0

Table 15: Referring-expression comprehension results on RefCOCOg and HumanRef. We report F1 at IoU 0.5 and F1 averaged across IoU thresholds. **Bold** and <u>underline</u> denote the best and second-best results, respectively, among vision-language models.

derlying the Avg. Refer. Exp. column. Per-benchmark dense-grounding results on COCO and LVIS, underlying the Avg. Dense Grounding column, are reported separately in Table 5 in the main text.

G Accuracy-Latency Comparison Details

This section provides details of Figure 1. We benchmark all models on an Apple M1 Max (32 GB) using MLX 0.32.2 and mlx-vlm 0.6.17, with 4-bit affine weight quantization (group size 64). After two warmup iterations, we report the median latency over 20 runs with explicit device synchronization. To approximate worst-case deployment cost, inputs are evaluated at fixed square resolutions, ensuring the maximum visual-token count for each operating point.

Latency evaluation. For the dense grounding experiments in Table 5, we report both prefill and end-to-end latency. We fix the maximum visual input budget to $589{,}824$ (0.59M) pixels for all models and measure prefill latency at this common pixel budget. End-to-end latency additionally depends on the cost of autoregressive decoding, which can vary substantially with the representation used to serialize bounding boxes. We therefore determine the token cost of each model's native grounding output format by querying the model and tokenizing the resulting box representation with its corresponding tokenizer. As summarized in Table 16, the resulting representations range from compact location tokens to text-based integer or floating-point coordinates.

To compare models under the same detection workload, we set the generation budget to seven boxes, corresponding to the average number of annotated boxes per image in the COCO validation set. The decode budget for each model is therefore its measured tokens per box multiplied by seven. Our quantized location representation requires 6 tokens per box, or 42 tokens for seven detections. In comparison, the text-based representations require 119–204 tokens for the same workload. Relative to these text-based formats, quantized coordinates reduce the decode-token budget by 64.7%–79.4%, with an average reduction of 72.2%, corresponding to $3.6\times$ fewer output tokens. End-to-end latency in Table 5 combines prefill latency at the common 0.59M-pixel input budget with decoding under these model-specific seven-box token budgets. This protocol therefore measures both encoder-side efficiency and the decoding cost imposed by each model's native grounding representation. For Figure 1(right), we use the same decode budget but vary the input pixels to obtain the various operating points.

Operating points. In Figure 1, each point on a curve is one input resolution, identified by its maximum number of input pixels, given in millions (M). We sweep inputs of 0.3M, 0.6M, 1.0M, 1.6M, and 2.4M pixels. All models except InternVL3.5 and MiniCPM-V-4.6 process images at native resolution, so their cost is set by the total pixel count; these two instead split the image into

Model	Coordinate representation	Tokens / box	Tokens / 7 boxes	Token reduction
FastVLM2	Quantized location tokens	6.0	42	–
Rex-Omni	Quantized location tokens	6.0	42	–
LocateAnything-3B	Quantized location tokens	6.0	42	–
MiniCPM-V-4.6	Text coordinates	17.0	119	64.7%
Qwen3-VL	Integer coordinates	20.0	140	70.0%
InternVL3.5	Text box serialization	20.3	142	70.4%
Ovis2.5-2B	Normalized floating-point coordinates	29.1	204	79.4%

Table 16: **Over 60% reduction in decode-token budget.** Decode-token budgets used for the dense grounding latency evaluation. Tokens per box are determined from each model's native output format, and the per-image budget assumes seven detections, the average number of annotated boxes in the COCO validation set. Savings are relative to our quantized location-token representation.

448^2 tiles (MiniCPM-V-4.6 does not appear in Figure 1). InternVL3.5 has no resolution setting and adds a thumbnail when there is more than one tile, so we sweep its tile budget instead, giving 1, 5, 10, and 17 tiles, or 0.2M to 3.4M pixels (256 to 4352 visual tokens). These sweeps differ from the settings used in Table 4, which evaluates each model with the default minimum and maximum pixel limits shipped with it; Figure 1 instead sets the maximum to 2.4M pixels at its largest operating point. Accuracy and latency can therefore differ slightly between the two, and the numbers in Figure 1 and Table 4 should not be compared directly.

Resolution range. In Figure 1, the VQA plot shows only operating points that produce at least 144 visual tokens. FastViTHD downsamples its input by $64\times$ per side, so FastVLM2 produces $N/64^2$ visual tokens for an input of N pixels: 64 tokens at 0.26M pixels and 144 at 0.59M. Every other model already exceeds 144 visual tokens at its smallest plotted input, so the threshold only removes FastVLM2 at 0.3M pixels. For grounding, the curves begin at 0.3M pixels but terminate at lower resolutions than the VQA curves because performance saturates earlier. The RefCOCO, RefCOCO+, and RefCOCOg images come from COCO and are at most 640 pixels on their longest side (at most 0.4M pixels), so a larger input budget adds no image detail. Beyond 1.0M pixels, accuracy changes by at most 0.2 points for every model we evaluated there (Qwen3-VL-4B, Rex-Omni, and Ovis2.5-2B). We therefore omit the flat 1.6M- and 2.4M-pixel points of Qwen3-VL-4B and Rex-Omni, and do not evaluate FastVLM2 beyond 1.0M pixels.