Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

VectorLLM++: A Unified Next-Token-Prediction MLLM for Dense Remote Sensing Perception and Vector Mapping

Shunping Ji

jishunping@whu.edu.cn

Wuhan University https://orcid.org/0000-0002-3088-1481

Article

Keywords: Remote sensing MLLM, Vector maps, Unified geometric tokenization, Dense perception

Posted Date: September 10th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10949232/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: There is **NO** Competing Interest.

VectorLLM++: A Unified Next-Token-Prediction MLLM for Dense Remote Sensing Perception and Vector Mapping

Abstract

Large-scale vector maps underpin diverse human activities, yet their automated generation is hindered by scarce vector-format training data and the lack of MLLMs that produce structured vector outputs. We present VectorLLM++, a unified MLLM that reformulates remote sensing perception as next-token prediction across seven tasks: polygon extraction, dense segmentation, dense detection, in-context detection, referring detection, change detection, and visual question answering. Central to our approach is the insight that a pixel mask and its vectorized boundary are two sides of the same object; we therefore introduce a vector tokenizer that discretizes coordinates into a 1024-token vocabulary and a mask tokenizer that compresses masks into codebook-based discrete tokens, enabling seamless integration of geometry and language within a single next-token-prediction framework. To address data scarcity, we develop a semi-automatic vector-instruction data engine that generates over 1.2 million polygon annotations. We train the model through large-scale pre-training on 62.9 million samples from 57 datasets, supervised fine-tuning on 7.2 million instructions, and reinforcement learning with verifiable rewards that directly optimize geometric quality. Extensive experiments on 23 datasets show that VectorLLM++ consistently outperforms previous state-of-the-art task-specialized models across all seven tasks. By unifying raster and vector representations, VectorLLM++ establishes a general-purpose framework for next-generation geospatial intelligence.

Keywords: Remote sensing MLLM, Vector maps, Unified geometric tokenization, Dense perception

1 Introduction

As a primary deliverable, large-scale vector maps produced from remote sensing imagery underpin a wide range of sectors involving trillions of dollars in annual economic activity, such as urban planning, engineering construction, cadastral surveying, and navigation, by virtue of offering a structured and measurable geometric representation of objects of interest in contrast to pixel-level semantic segmentation outputs [1–9]. Despite over three decades of research on automatic vector

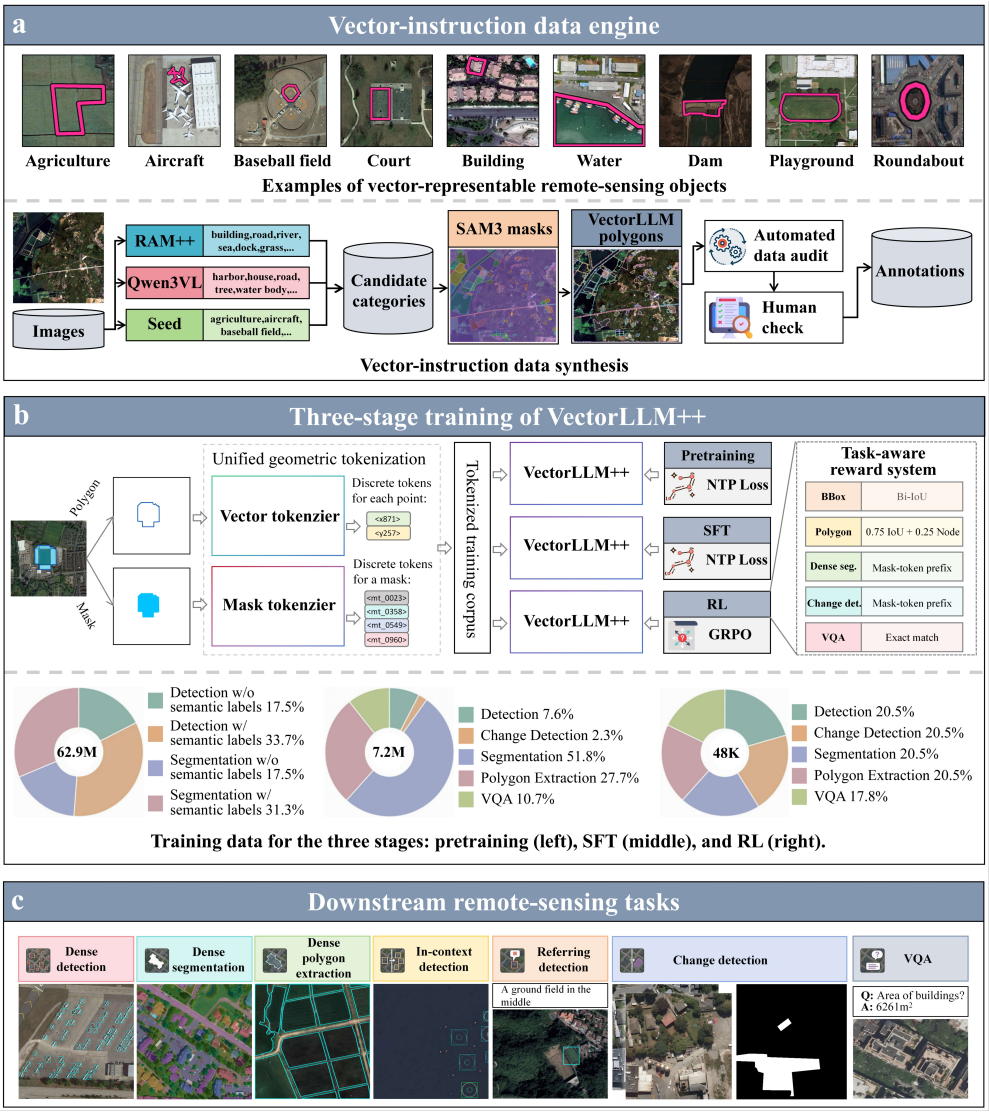


Fig. 1: Overview of VectorLLM++. **a**, Vector-instruction data engine operates through four core stages: semantic tagging, promptable segmentation, vector boundary generation, and quality assurance. **b**, VectorLLM++ comprises a base MLLM, a vector tokenizer, and a mask tokenizer. Leveraging the tokenized training corpus, we conduct pre-training and SFT under the unified next-token prediction paradigm, followed by reinforcement learning with verifiable rewards. To accommodate diverse remote sensing perception tasks during reinforcement learning, we design a task-aware reward system. **c**, VectorLLM++ supports seven downstream remote sensing perception tasks.

map generation, manual delineation remains the predominant approach in practice today, entailing substantial labor costs and inefficiencies [4–6]. Large language models (LLMs) and multimodal large language models (MLLMs) have recently emerged as the most promising tools for automating vector map generation [10–14]. Nevertheless, while the vast majority of remote sensing MLLMs are tailored for visual question answering (VQA) and localized pixel-level perception or non-generative earth observation tasks [15–23], only a handful exhibit dense pixel understanding, and VectorLLM stands as an exception designed for vector-oriented mapping [24]. In this work, we extend VectorLLM—originally designed for building mapping—to automatically produce regularized vector maps of general remote sensing objects that align with manual delineation. Furthermore, we take a significant step forward in dense pixel understanding, motivated by the insight that pixel-based mask and its vectorized boundary are two sides of the same object. To this end, we introduce a Remote Sensing Mask Tokenizer (RSM-Tok), which translates each mask into a compact sequence of discrete tokens—akin to a word in a visual codebook—that can be seamlessly consumed by a pure next-token-prediction LLM without any visual decoder [25–27]. This design not only eliminates the need for auxiliary visual decoders found in prior MLLMs [28–30], but also provides a unified token-based formulation that seamlessly bridges mask segmentation and vectorized boundary generation. Furthermore, our unified next-token-prediction architecture facilitates the adoption of reinforcement learning (RL) for post-training [31–34]. Unlike previous pixel-wise MLLMs, which require a mask decoder to convert hidden states into 2D masks for IoU-based reward computation, our token-based formulation enables direct reward assignment at the token level without any auxiliary decoding.

Beyond vector mapping, our MLLM—named VectorLLM++—is designed for general vision-language tasks such as VQA, referring detection, dense detection, and change detection [3, 15–19, 35]. Critically, we incorporate in-context learning to accommodate novel ground objects that inevitably emerge beyond the categories covered by existing open-source datasets. This provides an attractive alternative to the heavy and often impractical full-scale retraining that would otherwise be required for each new category [36, 37]. Such a capability is particularly crucial given the significant category gap: even when aggregating all existing open-source remote sensing datasets and merging synonyms, the total number of distinct object classes amounts to no more than 80, whereas a standard 1:2000-scale topographic map requires recognition of over 300 level-2 categories.

Building such a vector-oriented yet general-purpose remote sensing MLLM, however, entails three foundational efforts that go beyond simply extending an existing architecture. First, we construct a vector data engine to curate large-scale vector data for training, given the scarcity of existing vector datasets [4–6, 24]; in contrast, training data for other vision-language tasks can be largely sourced from open repositories [3, 15, 26, 35]. Second, we bridge the modality gap by encoding all output forms—language, bounding boxes, polygons, and masks—into a shared discrete token vocabulary. This unification ensures that the MLLM can handle vector mapping, VQA, detection, and segmentation within a single autoregressive framework, without the need for heterogeneous decoders. Third, we design tailored reward signals

for RL within our unified framework. While pre-training and supervised fine-tuning (SFT) provide a reasonable starting point, its outputs often suffer from missing components, over-segmentation, irregular zigzag boundaries, and poor alignment with human-drawn delineations—issues that RL with carefully designed token-level rewards can effectively mitigate.

1.1 Vector-instruction data engine

We construct a curated vector-instruction data engine using public and internal remote sensing datasets. The data engine integrates semantic tagging, promptable segmentation, vector boundary generation, and manual quality assurance, to produce dense vector annotations with broad category coverage. This data engine expands the scale and diversity of geometric supervision while exposing the model to the annotation conventions used in human cartographic production. Specifically, the pipeline operates in three stages. First, we generate rich image tags to describe candidate objects and their contextual cues using RAM++ [38] and Qwen3VL [14]. Second, we employ SAM3 [39] and VectorLLM [24] to generate initial vector candidates, which are subsequently refined through polygon simplification, quality filtering, and manual inspection. Third, we reformat the finalized annotations into instruction-following and in-context learning templates, allowing target objects to be specified via natural language, few-shot geometric exemplar, or a combination of both. Using the data engine, we create a huge open-vercabulary training dataset covering over one million samples with more than 1000 raw category tags, and a test benchmark containing 8,820 instances of 18 representative categories.

1.2 Unified geometric tokenization

To enable an MLLM to generate vector and other outputs within a unified next-token-prediction framework, we formulate a geometric tokenization scheme that encodes geometric primitives—bounding boxes, polygons, and segmentation masks—into a shared discrete token space, in contrast to prior approaches that treat different primitives with separate task-specific heads and decoding strategies [26–30]. This allows the model to treat geometric outputs as language-like sequences, eliminating the need for task-specific decoders. Specifically, we serialize bounding boxes and polygons as coordinate-token sequences under a normalized image-relative reference frame, which enhances robustness to variations in image size, crop scale, and ground sampling distance without requiring input resizing or re-scaling. For polygons, we further design a sequence grammar that explicitly models vertices, instance delimiters, polygon closure, and termination symbols, enabling the model to generate both single-instance and multi-instance vector maps in a unified manner. For segmentation masks, we further design a discrete mask tokenizer, RSM-Tok, that compresses dense regions into compact tokens within a learned visual vocabulary. These tokens can be processed directly by the language model and subsequently decoded back into pixel-level masks. With this unified formulation, different geometric primitives can be generated through a single multimodal interface, without requiring separate task-specific heads while preserving the structural properties unique to each representation.

1.3 Verifiable geometric reward design

While likelihood-based pre-training and SFT yield reasonable next-token predictions, they do not guarantee that the decoded geometries are complete, semantically correct, or structurally plausible—issues we identified earlier. To address these limitations, we adopt reinforcement learning with verifiable rewards (RLVR) as post-training and design a task-aware reward system tailored to dense remote sensing mapping. Building on our unified token-based formulation, which enables direct token-level reward assignment without auxiliary decoders, we define rewards that explicitly penalize missing objects, semantic confusion, invalid geometry, over-fragmentation, and deviation from human delineation standards. We compute rewards automatically from parsed model outputs and reference annotations, providing scalable supervision without relying on a learned reward model. Our reward design targets the geometric requirements of three complementary output modalities. For bounding boxes, we emphasize semantic grounding, localization precision, and object completeness. For polygons, we reward region-level alignment while encouraging compactness and editability. For segmentation masks, we provide a verifiable token-level consistency reward for mask generation. In addition, the reward system explicitly accounts for output parseability and format validity, directly penalizing malformed or unrecoverable geometric predictions. More importantly, this task-aware reward design aligns the model with mapping-oriented criteria that likelihood-based training alone fails to capture. Ultimately, by transforming geometric correctness into verifiable feedback, it improves fundamental capabilities—phrase grounding and spatial accuracy—while also enhancing output quality in terms of validity, compactness, and editability.

Building on these three components, we propose VectorLLM++, an 8-billion-parameter MLLM designed for general remote sensing object vector mapping. To our knowledge, it is the first model that unifies vector boundary and mask representations within a shared discrete token space, while also supporting a broad range of tasks—including VQA, referring/dense detection, change detection, and in-context learning—under a consistent next-token-prediction paradigm. For training, we prepared one million vector-instance annotations across diverse object categories using our data engine (Fig. 1a), alongside publicly available 56 datasets for other vision-language tasks. The model is trained in three stages, as shown in Fig. 1b. Pretraining uses 62.9 million samples covering both remote sensing and close-range imagery. SFT is performed on 7.2 million remote sensing samples. Finally, reinforcement learning with verifiable rewards (RLVR) is applied on 48,000 samples to further refine geometric alignment.

As shown in Fig. 1c, we evaluate VectorLLM++ on 23 typical Earth observation (EO) datasets, covering seven tasks across domains such as crop census, land surveying, water conservancy, urban planning, traffic engineering, and earthquake-induced change detection. This evaluation establishes a large-scale, multi-dimensional benchmarking framework for remote sensing MLLMs, with a particular focus on vector mapping—a key capability that has been largely overlooked in both existing datasets and remote sensing MLLMs. Compared to the best specialized MLLMs tailored for each task, VectorLLM++ achieves consistent improvements with a single unified next-token-prediction architecture: +25.7 pp F1 in dense detection, +13.3 pp mAP in

polygon mapping, and +33.5 F1 in in-context learning; +24.4 F1 in referring detection, along side +3.0 pp mIoU in change detection, +6.5 pp mIoU in segmentation, and +3.0 pp OA in VQA. These substantial enhancements fill a critical gap in modern remote sensing MLLMs by enabling vector mapping—a capability that has been largely absent—while also improving mask prediction and other vision-language tasks. These advances benefit a wide range of real-world applications, including urban planning, engineering construction, disaster monitoring, and agricultural management, among others [1, 2]. Overall, VectorLLM++ provides insight into how to design a modern remote sensing MLLM under a unified next-token-prediction paradigm that integrates diverse geometric and semantic outputs, takes a firm step forward for automated human-aligned vector mapping that has troubled us for more than thirty years, and lays a solid foundation for next-generation geospatial intelligence systems.

2 Results

We present VectorLLM++, a remote sensing MLLM capable of seven tasks, including vector mapping and mask segmentation. The mask tokenizer RSM-Tok is independently trained for 3 days on 128 H20 GPUs and then totally frozen in VectorLLM++. The main model undergoes pre-training for 14 days on 128 H20 GPUs, followed by SFT for 10 days and RL post-training for 5 days on 8 H100 GPUs. VectorLLM++ supports native-resolution image input, preserving original image resolution throughout training and inference without task-specific cropping or resizing. For evaluation, we construct a benchmark that combines public EO datasets with our purpose-built GeoPolygon dataset.

2.1 Benchmark for seven tasks

The benchmark is designed to evaluate whether a single model can produce multiple forms of structured geospatial outputs—specifically, bounding boxes, polygons, segmentation masks, and text answers—across seven task types: dense detection, dense segmentation, dense polygon extraction, in-context detection, referring detection, change detection, and VQA, drawing on 23 Earth observation datasets in total. These tasks differ in their granularity: dense understanding tasks require holistic scene interpretation, whereas sparse tasks such as referring detection focus on specific objects or regions rather than full-image processing. Detailed dataset statistics and metric definitions are provided in the supplementary material. For dense polygon extraction, we introduce GeoPolygon, a multi-category benchmark for evaluating vector contour generation. It comprises 1,452 unique images, 8,820 polygon instances, and 18 object categories. Unlike detection and segmentation datasets, GeoPolygon provides polygonal contours delineated by human experts according to cartographic conventions as ground-truth annotations, enabling direct evaluation of vectorized contour generation.

2.2 VectorLLM++ achieves superior performance on all tasks.

Table 1 reports the main results across all tasks on the benchmark. VectorLLM++ after SFT outperforms the previous best method across all 23 reported datasets and

Table 1: Overall performance on the benchmark. We compare VectorLLM++ with the strongest previously reported method for each dataset and report the SFT and RL checkpoints. Scores are percentages, and higher values indicate better performance for all metrics. Bold values denote the best results, and underlined values denote the second-best results.

Task type	Dataset	Metrics	Previous best		VectorLLM++	
			Method	Score	SFT	RL
Dense detection	DIOR	F1(↑)	RSCoVLM	41.0	<u>58.4</u>	**60.9**
	HRSC2016	F1(↑)	Rex-Omni	43.4	<u>71.0</u>	**72.3**
	WHU	F1(↑)	Rex-Omni	45.0	<u>70.2</u>	**71.2**
	WHU-Mix test1	F1(↑)	Rex-Omni	28.5	<u>40.1</u>	**60.2**
	WHU-Mix test2	F1(↑)	Rex-Omni	33.0	<u>42.0</u>	**55.0**
Dense segmentation	Globe230k	mIoU(↑)	Falcon	52.1	<u>60.4</u>	**60.7**
	SIOR	mIoU(↑)	UniGeoSeg	70.0	<u>75.3</u>	**76.0**
	SOTA	mIoU(↑)	UniGeoSeg	49.5	<u>57.9</u>	**58.2**
	LoveDA	mIoU(↑)	Falcon	43.5	<u>45.5</u>	**46.3**
Dense polygon extraction	GeoPolygon	mAP(↑)	VectorLLM	23.5	<u>35.5</u>	**36.8**
In-context detection (k=1)	MAC	F1@50(↑)	Rex-Omni	80.0	<u>86.3</u>	**88.7**
	Oilwell	F1@50(↑)	Rex-Omni	38.4	<u>62.9</u>	**86.1**
	SolarPanel	F1@50(↑)	Rex-Omni	15.6	<u>43.7</u>	**59.6**
Referring detection	DIOR-RSVG val	Acc@0.5(↑)	GeoGround	77.2	<u>85.4</u>	**86.9**
	DIOR-RSVG test	Acc@0.5(↑)	GeoGround	77.7	<u>85.9</u>	**86.3**
	RSVG val	Acc@0.5(↑)	GeoGround	27.6	<u>62.1</u>	**64.2**
	RSVG test	Acc@0.5(↑)	GeoGround	26.6	<u>67.6</u>	**69.1**
Change detection	WHU-CD	mIoU(↑)	Falcon	53.1	<u>60.5</u>	**61.1**
	SYSU-CD	mIoU(↑)	Falcon	56.1	<u>56.2</u>	**56.7**
	CCD	mIoU(↑)	Falcon	42.7	<u>43.0</u>	**43.2**
VQA	HR test	OA(↑)	SHRNet	92.1	<u>95.1</u>	**95.5**
	HR test_phili	OA(↑)	RSLVLM	93.1	<u>95.8</u>	**96.1**
	LR	OA(↑)	EarthDial	92.7	<u>95.1</u>	**95.2**

metrics, indicating that the unified instruction tuning is effective across different output formats. The reinforcement learning further improves VectorLLM++ on all 23 EO datasets. The same progression is visible in Fig. 2, where SFT produces cleaner non-VQA outputs than the previous SOTA and RL further improves alignment with the ground truth.

Dense detection. VectorLLM++ improves dense detection performance on all five evaluated datasets. Averaged over DIOR [3], HRSC2016 [40], WHU [41], and the two

WHU-Mix test splits [42], the previous best [43, 44] F1-score is 38.2, while VectorLLM++ reaches 56.3 after SFT and 63.9 after RL. The gains are particularly large on the more challenging WHU-Mix splits: on WHU-Mix test1, the RL checkpoint improves F1 from 28.5 to 60.2, and on WHU-Mix test2 from 33.0 to 55.0. On the standard DIOR, HRSC2016, and WHU benchmarks, VectorLLM++ also improves the previous best scores by 19.9, 28.9, and 26.2 points, respectively. These results show that the model can generate dense box outputs reliably across object categories and dataset domains.

Dense segmentation. VectorLLM++ consistently outperforms previous specialist models [45, 46] on all four benchmarks. The average mIoU increases from 53.8 for the previous best methods to 59.8 after SFT and 60.3 after RL. The largest improvements are observed on Globe230k [47] and SOTA [48], where the RL checkpoint improves mIoU by 8.6 and 8.7 points, respectively. On SIOR [48] and LoveDA [49], VectorLLM++ reaches 76.0 and 46.3 mIoU, improving the previous best results by 6.0 and 2.8 points. The relatively small but consistent RL gains over SFT suggest that SFT already on massive training data accounts for most of the segmentation improvement, while the RL stage provides additional refinement.

Dense polygon extraction. GeoPolygon evaluates a strict vector-output setting, where the model must produce explicit polygon contours rather than boxes or raster masks. On this benchmark, VectorLLM++ improves mAP from 23.5 (VectorLLM [24]) to 35.5 after SFT and 36.8 after RL. The 13.3-point gain over VectorLLM demonstrates that the geometry-instruction data and the polygon-oriented training pipeline substantially improve contour-level localization.

In-context detection. VectorLLM++ substantially improves one-shot visual prompting performance across all three evaluated categories. The average F1@50 increases from 44.7 (Rex-Omni [44]) to 64.3 after SFT and 78.1 after RL. The RL checkpoint also raises MarineAquacultureCages (MAC) from 80.0 to 88.7. The gains are particularly large on Oilwell and SolarPanel, where F1@50 improves by 47.7 and 44.0 points over the previous best results. These results indicate strong category transfer from the visual prompt to dense box prediction.

Referring detection. VectorLLM++ achieves consistent improvements on both DIOR-RSVG [50] and RSVG [50]. On DIOR-RSVG, the RL checkpoint achieves 86.9 Acc@0.5 on the validation split and 86.3 on the test split, exceeding the previous best method, GeoGround [19], by 9.7 and 8.6 points. The gains are larger on RSVG, where expressions are longer and more spatially descriptive: VectorLLM++ improves Acc@0.5 from 27.6 to 64.2 on the validation split and from 26.6 to 69.1 on the test split. Compared with the SFT checkpoint, RL further improves all four splits, with gains ranging from 0.4 to 2.1 points.

Change detection. VectorLLM++ improves mIoU on all three bi-temporal change detection benchmarks. The RL checkpoint achieves 61.1 on WHU-CD [41], 56.7 on SYSU-CD [51], and 43.2 on CCD [52], increasing the average mIoU from 50.6 (previous best) to 53.7. The improvement is largest on WHU-CD (+8.0 points), while SYSU-CD and CCD show smaller but consistent gains, suggesting robust mask generation for changed regions.

VQA. VectorLLM++ achieves consistently high overall accuracy (OA) on all three splits [35]. The RL checkpoint obtains 95.5 on HR test, 96.1 on HR test_phili, and 95.2 on LR, improving the average OA from 92.6 (previous best [53, 54]) to 95.6. The consistent gains over SFT show that the RL stage also benefits simple remote sensing question answering.

2.3 VectorLLM++ delivers superior polygon extraction across diverse remote sensing objects

A realistic dense polygon extraction strategy requires two coupled capabilities: open-vocabulary remote sensing object detection and polygon extraction. GeoPolygon therefore provides a direct test of whether a model can localize category-specified objects and represent their boundaries as explicit polygons. While VectorLLM [24] does not natively support open-vocabulary detection and relies on an external detector to first localize objects and then extracts polygons within those detected boxes, VectorLLM++ performs category-conditioned object localization and polygon generation within a single MLLM framework. Fig. 3a shows the category-level mAP gains on GeoPolygon. VectorLLM++ improves polygon extraction for diverse remote sensing objects and scenes, including vehicles, chimneys, stadiums, water bodies, buildings, playgrounds, roundabouts, covering both man-made and natural land-cover classes with diverse sizes and densities. The largest gains after SFT are observed for vehicle (+41.2), chimney (+41.1), stadium (+25.8), water (+24.0), and buildings (+23.3). The overall comparison in Fig. 3c further demonstrates the benefit of integrating open-vocabulary detection with polygon generation. RSCoVLM [43]+VectorLLM [24] obtains 21.7 mAP, while Rex-Omni [44]+VectorLLM [24] obtains 23.5 mAP. VectorLLM++ reaches 35.5 mAP after SFT and 36.8 mAP after RL, improving over the stronger Rex-Omni+VectorLLM baseline by 13.3 points. This result indicates that the gains are not only due to replacing the detector, but also reflect the end-to-end dense polygon extraction capability learned by the unified VectorLLM++. In addition to the open-vocabulary localization capacity, VectorLLM++ benefits from large-scale bounding-box, segmentation mask, and polygon supervision, which jointly expose the model to multiple geometric output formats. We also evaluate the generated polygons from the perspective of vector shape quality. As shown in Fig. 3d, VectorLLM++ substantially improves polygon IoU and C-IoU over VectorLLM. The polygon IoU increases from 49.9 to 70.0, and C-IoU increases from 41.3 to 56.8, indicating better overlap and complexity-aware contour alignment. For N-ratio, a metric that measures point-wise contour fidelity, VectorLLM++ increases the score from 61.4 to 79.7, indicating better consistency with human-aligned point-by-point delineation. These results reflect that VectorLLM++ improves both object-level matching and vector contour quality, though polygon complexity control remains an orthogonal challenge to mAP optimization. Moreover, VectorLLM++ supports in-context detection, allowing visual exemplars to specify target categories on the fly. We evaluate this ability under both single-image and cross-image settings. As shown in Fig. 3e and 3f, increasing the number of visual exemplars improves in-context detection performance. In the single-image setting, F1@50 increases from 60.9 with k=1 to 71.7 with k=4, and mIoU increases from 26.0 to 40.9. In the cross-image setting, F1@50 increases from

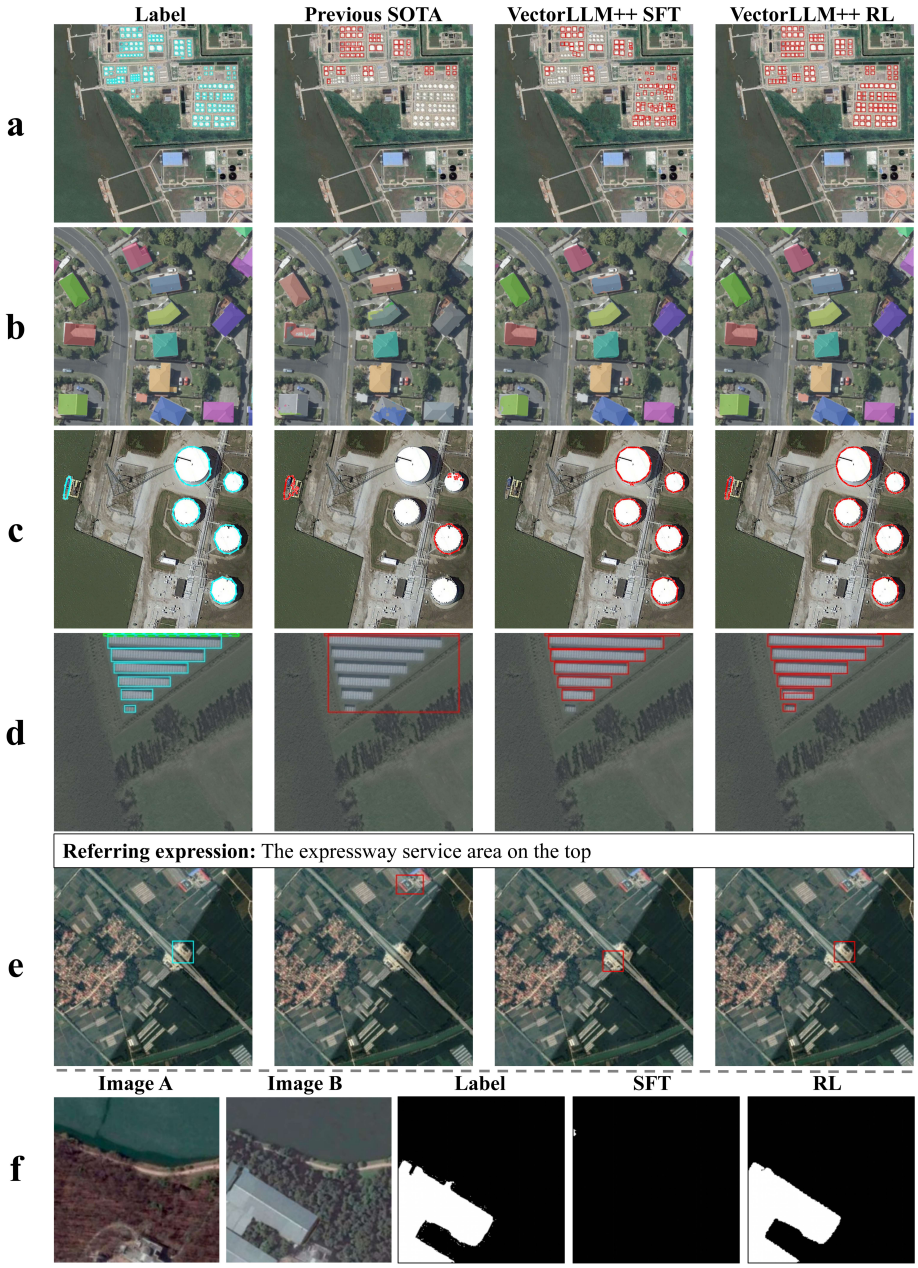


Fig. 2: Visual comparison of the previous SOTA, VectorLLM++ SFT and RL checkpoints across various tasks. **a**, Dense detection. **b**, Dense segmentation. **c**, Dense polygon extraction. **d**, In-context detection, with green bounding boxes indicating the provided visual exemplars. **e**, Referring detection. **f**, Change detection.

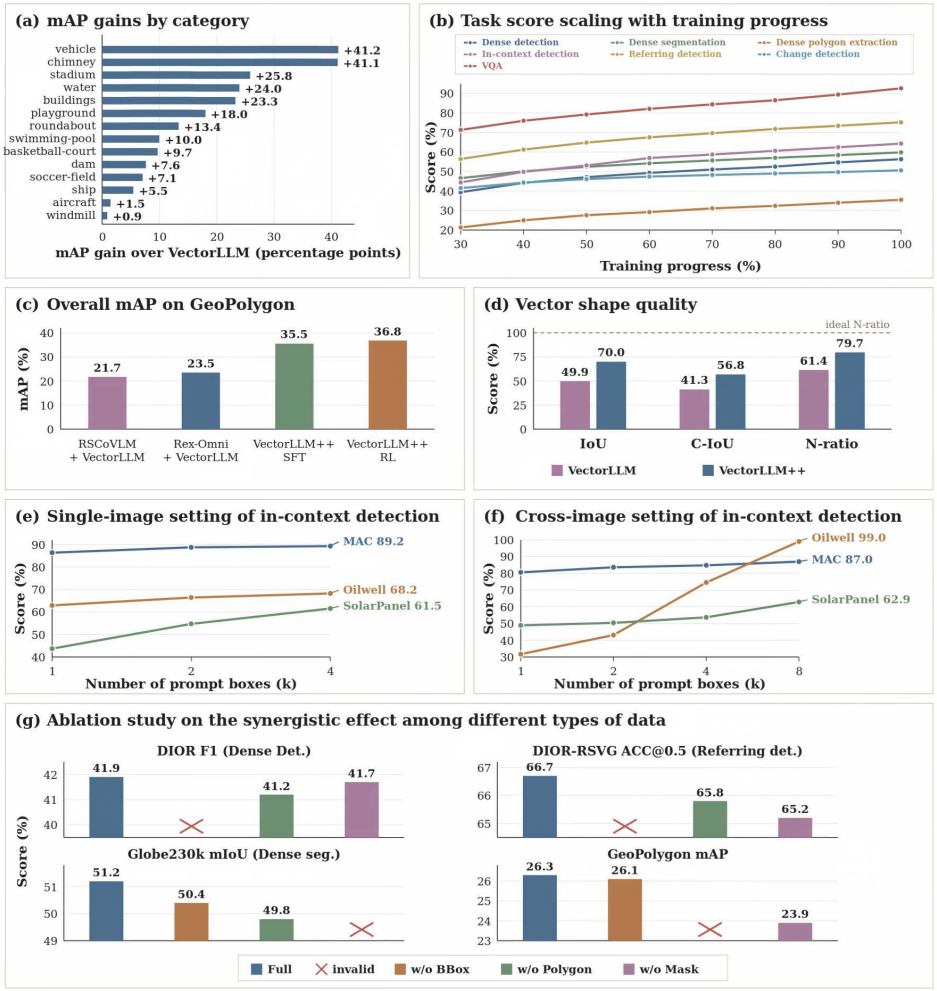


Fig. 3: Additional quantitative analysis of VectorLLM++ performance. **a**, Category-wise mAP improvements on GeoPolygon compared with VectorLLM. **b**, Task score trajectories across training steps for multiple tasks. **c**, Overall GeoPolygon mAP comparison among baseline combinations and VectorLLM++ variants. **d**, Vector shape quality evaluated via IoU, C-IoU and N-ratio. **e**, Single-image in-context detection performance as the number of prompt boxes increases. **f**, Cross-image in-context detection performance with increasing prompt boxes. **g**, Ablation study on the synergistic effect among different types of data. "Full" denotes training with all types of data; "invalid" indicates the model fails to generate valid outputs and thus cannot be scored, owing to the absence of task-specific data; "w/o" stands for prediction results without the corresponding type of data in the training set. Training with all types of data (blue bars) yields the highest performance score on every task.

52.6 with k=1 to 73.9 with k=8, and mIoU increases from 27.3 to 37.3. The recall gains are especially clear in both settings, rising from 60.6 to 81.7 in the single-image setting and from 57.1 to 82.4 in the cross-image setting. These results show that VectorLLM++ can use visual exemplars from different images as category prompts, extending its open-vocabulary detection capability beyond fixed training categories.

3 Discussion

3.1 Is there a synergistic effect among different types of dense perception data?

A central motivation of VectorLLM++ is that a pixel-based mask and its vectorized boundary represent two sides of the same object. Along with bounding boxes for localization, these three geometric primitives are unified under a single next-token-prediction architecture. Learning them jointly not only expands the range of supported tasks, but also offers a pathway toward a more unified geometric representation of remote sensing objects. Beyond this conceptual advantage, the insight is substantiated by our experimental results. The data-ablation results in Fig. 3g suggest that dense perception data should not be viewed as a collection of isolated output grammars; rather, bounding boxes, masks, and polygons introduce complementary priors. Removing one annotation type, therefore, has two effects: it disables the corresponding output grammar and, without complementary supervision is absent, can also weaken performance on the remaining valid tasks. This observation carries implications for scaling dense remote sensing models. Increasing the total amount of supervision is important, but data composition is at least as critical. A dataset dominated by a single annotation type may fail to provide the geometric constraints needed for other output formats. Future work should therefore investigate the ratio, curriculum, and co-training schedule of box, mask, and polygon data, rather than treating scale alone as the primary variable.

3.2 Towards vector mapping of more object categories

The large number of ground object categories presents a major obstacle for automated mapping, with many categories being rare in practice. As an initial step, we establish an 18-category vector mapping benchmark in this work. To handle rare categories beyond this set, we introduce in-context learning with visual exemplars. This offers a complementary pathway to open-vocabulary models: rather than relying on text–image alignment, a visual exemplar specifies the target category directly in the visual feature space shared with the query image, preserving appearance, scale, and contextual cues. The ablation studies on exemplar number (Fig. 3e and Fig. 3f) show that visual prompts consistently improve detection performance, even in cross-image settings, suggesting that VectorLLM++ can transfer category concepts from reference images to unseen scenes. Nevertheless, several limitations remain. First, the knowledge acquired from in-context exemplars is only available during inference and is not consolidated into the model's parametric memory. A promising direction for future work is to incorporate few-shot learning to update a subset of model parameters,

enabling the model to retain knowledge about new categories without full retraining. Second, certain categories are inherently ambiguous in both visual appearance and semantic definition—for instance, sparsely vegetated land versus bare soil or grassland. This ambiguity also arises in human annotation, where different experts may apply conflicting criteria. A promising direction is to incorporate human-aligned verification, such as multi-annotator consensus or active learning with expert feedback, to resolve ambiguous cases and establish clearer classification boundaries.

3.3 How should data scaling be allocated to achieve balanced improvements across tasks?

High-quality open-source remote sensing datasets—especially vector annotations—remain far less abundant than their natural-image counterparts. As long as this data scarcity persists, improvements in dataset size, quality, and diversity will continue to drive MLLM performance. As shown in Fig. 3b, all tasks' capabilities of the model improve with increasing data volume. This is especially true for knowledge-intensive tasks such as VQA, which are more sensitive to data size. Nevertheless, for dense polygon extraction, simply adding more examples is not enough—training must also incorporate explicit surveying and mapping conventions. This requires greater emphasis on geometric fidelity—including boundary generalization, vertex placement, and topology consistency—while simultaneously expanding data diversity to cover small, thin, occluded, or densely packed instances. For change detection, the key bottleneck likely lies in temporal pair sampling, negative–positive balance, and real-world scene variation. More broadly, future extensions of VectorLLM++ should jointly consider sampling ratios, annotation quality, tokenizer capacity, and curriculum schedules—i.e., training strategies that progressively increase instance difficulty—so that additional data translates into balanced improvements across diverse tasks rather than reinforcing only the dominant data sources.

4 Methods

4.1 Vector-instruction data engine

To address the scarcity of vector-level training data, we design a semi-automatic vector-instruction data engine that converts remote sensing image tiles into verified object-level vector annotations. As illustrated in Fig. 1a, given an input tile, the pipeline identifies candidate object concepts, generates class-conditioned masks, converts the masks into vector polygons, and finally verifies geometric consistency between the raster and vector representations. Only annotations that pass automatic validation or are corrected by human annotators are retained. This design decouples semantic concept discovery from geometric boundary extraction, enabling high category coverage while maintaining mapping-grade annotation quality.

Candidate object discovery. For each image tile I, we construct an open-vocabulary set of candidate object tags from three complementary sources. First, we apply RAM++ [38] to the image to obtain high-recall semantic tags. These tags serve as coarse object- and scene-level cues, providing robust fallback coverage for broad

categories such as buildings, roads, water bodies, vegetation, vehicles, and bare land. Second, we prompt Qwen3-VL [14] to generate a detailed caption of the image. We parse object-related noun phrases and category expressions from the caption, which typically offer more fine-grained and context-specific concepts than the RAM++ tags, for example, industrial tanks, temporary shelters, greenhouse structures, parking lots, sports fields, or construction areas. Third, we introduce a manually curated candidate vocabulary covering the seed mapping categories and their common aliases. This fixed vocabulary ensures that important target classes are not missed when they are visually subtle, poorly described by the caption, or absent from generic tagging vocabularies. The three tag sources are merged into a unified candidate set,

$$\mathcal{T}(I) = \mathrm{Merge}\big(\mathcal{T}_{\mathrm{RAM}}(I), \mathcal{T}_{\mathrm{cap}}(I), \mathcal{T}_{\mathrm{manual}}\big), \tag{1}$$

where $\mathcal{T}_{\mathrm{RAM}}$ denotes RAM++ tags, $\mathcal{T}_{\mathrm{cap}}$ denotes categories parsed from the Qwen3-VL caption, and $\mathcal{T}_{\mathrm{manual}}$ denotes the manually specified candidate vocabulary. The merge operation normalizes category names, removes duplicates, and resolves near-synonymous expressions into canonical labels while preserving useful aliases for prompting. We intentionally retain both coarse and fine-grained labels when they are semantically complementary: coarse tags improve recall, while fine-grained tags improve the specificity of downstream promptable segmentation.

Class-conditioned mask synthesis. The merged tag set $\mathcal{T}(I)$ is then used to guide promptable object segmentation. For each candidate tag $t \in \mathcal{T}(I)$, the image and the corresponding text prompt are fed to SAM3 [39] to generate a set of candidate object masks,

$$\mathcal{M}_t(I) = \{m_i^t\}_{i=1}^{N_t} \tag{2}$$

yielding dense, class-conditioned raster proposals for all candidate objects identified in the semantic discovery stage. Since multiple prompts may refer to the same physical object, the raw masks may contain duplicates across semantically related categories. We therefore apply mask-level non-maximum suppression based on pairwise mask IoU. When two masks exhibit large spatial overlap, the lower-confidence duplicate is suppressed, and their associated category tags are merged into a shared semantic label set. This allows the same object to inherit information from multiple prompts, for example, when a region is detected as both a building and an industrial facility. The resulting consolidated mask set is denoted as:

$$\widehat{\mathcal{M}}(I) = \{(\hat{m}_k, \hat{c}_k)\}_{k=1}^{K}, \tag{3}$$

where $\hat{m}_k$ is a retained object mask and $\hat{c}_k$ is its associated category descriptor. Each retained mask is converted into a tight bounding box,

$$\hat{b}_k = \mathrm{Box}(\hat{m}_k), \tag{4}$$

which is used to crop an object-centric image patch from the original tile. To provide sufficient boundary context, the crop is taken around the mask bounding box with a small spatial margin, while the coordinate transform between the crop and the

original image is recorded. This object-centric formulation reduces interference from neighbouring instances and aligns with the input format of VectorLLM [24], which is designed to extract the vector contour of the most salient central object in a cropped image.

Vector contour extraction. For each object crop, we use VectorLLM to generate a polygon for the central object. Let P_k^{crop} denote k-th polygon predicted in the crop coordinate system. We map it back to the original image tile using the stored crop transform:

$$P_k = \mathcal{A}_k^{-1}(P_k^{\text{crop}}), \tag{5}$$

where $\mathcal{A}_k$ is the affine transformation from the original image to the crop. The resulting polygon P_k is then post-processed to satisfy basic vector-map constraints: we close polygon rings, remove degenerate vertices, discard spurious small components, and simplify redundant boundary points using a controlled tolerance. This simplification reduces annotation complexity while preserving the overall object shape, yielding compact polygons that are easier to edit and integrate into GIS workflows. Finally, each geometric entity contains a polygon P_k, its corresponding SAM3 mask $\hat{m}_k$, and the merged category descriptor $\hat{c}_k$,

$$\mathcal{P}(I) = \{(P_k, \hat{m}_k, \hat{c}_k)\}_{k=1}^{K}, \tag{6}$$

where the mask provides a dense region reference, while the polygon offers a cartographic representation suitable for downstream vector-based training and evaluation.

Geometric consistency verification. While SAM3 masks and VectorLLM polygons provide complementary supervision, they may disagree when the segmentation mask is incomplete, the crop contains adjacent objects, or the polygon extractor follows a visually salient but semantically incorrect contour. To identify such cases, we rasterize each polygon P_k back into a binary mask $r(P_k)$ and compute its intersection-over-union score with the corresponding SAM3 mask:

$$\text{IoU}_k = \frac{|r(P_k) \cap \hat{m}_k|}{|r(P_k) \cup \hat{m}_k|}. \tag{7}$$

If $\text{IoU}_k \geq 0.8$, the polygon is considered geometrically consistent with the mask and is accepted after basic validity checks. If $\text{IoU}_k < 0.8$, the annotation is flagged as unreliable. The flagged cases are routed to human annotators for detailed inspection and repair. During review, annotators compare the original image, the SAM3 mask, and the VectorLLM polygon, then correct the boundary, adjust the category label, split or merge instances, or discard the sample if the target object is ambiguous. This verification strategy concentrates manual effort on high-risk cases rather than all generated annotations. By doing so, the pipeline retains the scalability of automatic segmentation and vector extraction while enforcing mapping-grade contour quality.

Instruction data synthesis. The verified annotations are finally converted into multimodal instruction-following samples. We unify these vector samples with publicly available data for other tasks into three general types of examples. First, language-prompted samples instruct the model to recognize all objects matching a textual category description and output them as bounding boxes, polygons, or masks. Second,

visual in-context samples provide one or more exemplar geometries and require the model to identify other objects of the same category. Third, mixed-prompt samples pair natural-language descriptions with exemplar boxes, polygons, or masks, teaching the model to interpret joint textual and visual specifications. The final data records are organized as tuples (I, q, y), where I is the image tile, q is the prompt (language, visual, or combined), and y is the target geometric output–a set of boxes, polygons, or segmentation masks depending on the task. This unified representation enables the model to learn dense mapping from heterogeneous prompts while sharing supervision across localization, vector extraction, and mask generation.

4.2 Model architecture

As illustrated in Fig.4a, VectorLLM++ consists of a base MLLM, a vector tokenizer, and a mask tokenizer. The base MLLM builts on Qwen3-VL [14] without modifying its original architecture, enabling native-resolution, multimodal, and multi-image inputs. To equip the base MLLM with diverse EO capabilities under the unified next-token prediction paradigm, we introduce a unified geometric tokenization scheme. Specifically, a vector tokenizer encodes geometric primitives—points, bounding boxes, and polygons–while a mask tokenizer encodes segmentation masks. During training, the two tokenizers convert original geometric annotations into discrete token sequences; during inference, they decode the predicted geometric token sequences back into coordinate sequences or masks, respectively.

Vector tokenizer. Bounding boxes and polygons are both composed of coordinate points; their floating-point values can in principle be represented as text sequences. However, directly using floating-point coordinates introduces several issues. First, a text tokenizer typically decomposes a floating-point number into multiple tokens rather than treating it as a single semantic unit, creating a misalignment between coordinate tokenization and the geometric representation expected by the model. Second, predicting floating-point values increases the difficulty of coordinate learning and complicates optimization. Third, remote sensing images vary in resolutions and aspect ratio, and objects differ substantially in size; both factors cause large variations in coordinate distributions and further complicate model optimization. To address these issues, we introduce a vector tokenizer that quantizes x- and y-axis coordinates into discrete tokens. Specifically, x-axis tokens span `<x0000>` to `<x1023>`, and the y-axis tokens span `<y0000>` to `<y1023>`. For example, `<x0000><y0000>` maps to the upper-left corner, while `<x1023><y1023>` maps to the lower-right corner. After tokenization, each coordinate value is represented as a single discrete token rather than being split into multiple text tokens. The coordinate distribution is normalized to the range of 0–1023, reducing optimization difficulty and improving generalization to unseen aspect ratios, resolutions, and object sizes.

Mask tokenizer. As shown in Fig. 4b, the mask tokenizer RSM-Tok consists of three coupled modules: a mask encoder, a quantizer, and a mask decoder. The mask tokenizer represents a binary object mask as a short sequence of quantized discrete mask tokens, enabling autoregressive mask-token generation in a MLLM followed by decoding back to a segmentation mask with the mask encoder. The mask encoder reuses the prompt pathway of SAM2 [55]. Specifically, the input mask is first resized

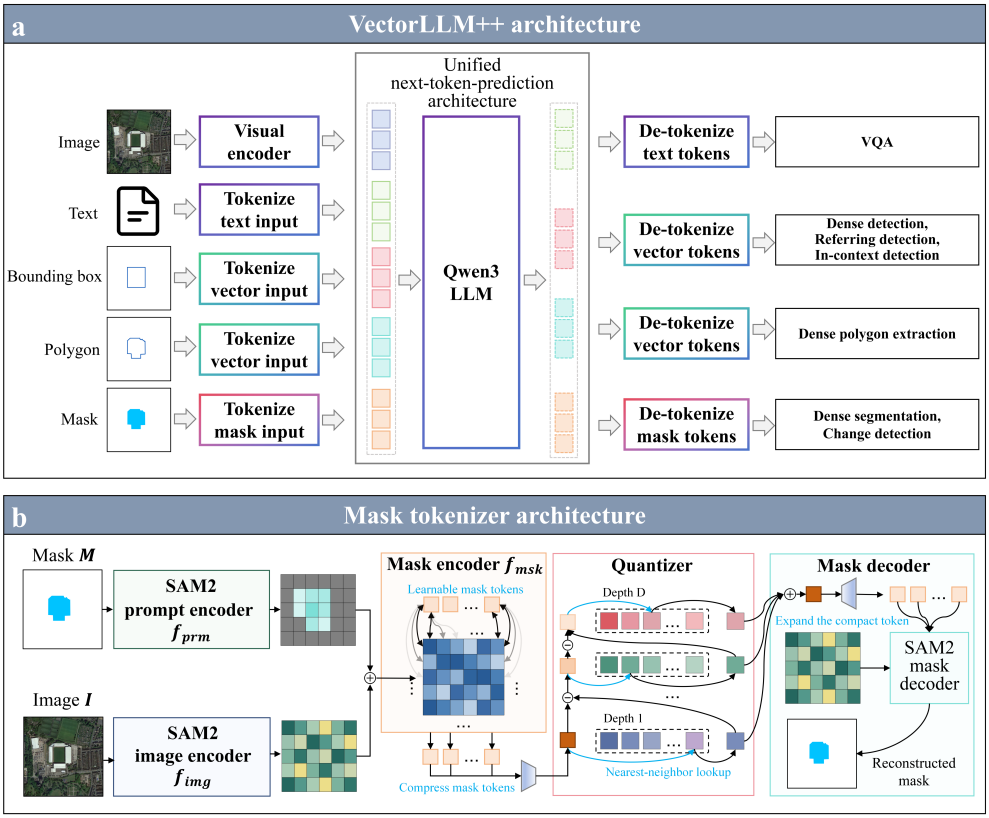


Fig. 4: Model architecture of VectorLLM++. **a**, VectorLLM++ consists of a base MLLM, a vector tokenizer and a mask tokenizer. The MLLM is built based on Qwen3VL. The vector tokenizer is responsible for tokenizing bounding boxes and polygons in training data into discrete geometric tokens, as well as de-tokenizing geometric tokens generated by the model back into bounding boxes and polygons. The vector tokenizer requires no modular design and can be implemented with just a few lines of code. The mask tokenizer tokenizes segmentation masks in training data into discrete mask tokens, and de-tokenizes mask tokens output by the model into segmentation masks. **b**, The mask tokenizer is mainly composed of a mask encoder, a quantizer and a mask decoder, and it reuses the prompt encoder, image encoder and mask decoder of SAM2.

to the SAM2 prompt encoder's required resolution and then fed into it as a mask prompt. The SAM2 prompt encoder outputs dense prompt embeddings, which are jointly fed with SAM2 image features into the mask encoder for fusion and further processing. The mask encoder uses a lightweight two-way transformer with learnable mask queries. Its main function is to generate K consecutive mask embeddings:

$$\mathbf{E} = f_{\text{msk}}\left(f_{\text{img}}(\mathcal{I}), f_{\text{prm}}(\mathcal{M})\right) = [e_1, ..., e_K],\ e_i \in \mathbb{R}^C, \tag{8}$$

where C denotes the feature dimension of each mask embedding and is set to 256 by default. Inside the two-way transformer, we use causal self-attention, ensuring that the embedding sequence respects autoregressive order: later embeddings can attend to earlier ones, while earlier embeddings cannot access information from later ones. This aligns the mask embedding sequence with the causal with constraint required by next-token prediction. We build RSM-Tok on SAMTok [56], which uses residual quantization to discretize consecutive mask embeddings, and introduce an additional bottleneck before quantization to make the representation more compact. Specifically, a group of K consecutive mask embeddings are first concatenated along the channel dimension and then compressed by an MLP into a single high-dimensional compact mask embedding:

$$\mathbf{e}_{\text{compact}} = \text{MLP}\big(\text{Concat}(\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_K)\big) \in \mathbb{R}^D, \tag{9}$$

where D denotes the dimension of the compact and is set to 512 by default. The compact embedding is then quantized through a sequence of residual quantization stages:

$$\begin{aligned} r_0 &= \mathbf{e}_{\text{compact}}, \\ z_d &= \arg\min_j \|r_{d-1} - c_{d,j}\|_2^2, \\ q_d &= c_{d,z_d}, \\ r_d &= r_{d-1} - q_d, \\ q &= \sum_d q_d. \end{aligned} \tag{10}$$

Here, r_d is the residual after the d-th quantization stage, $c_{d,j}$ is the j-th embedding in the codebook of the d-th stage, z_d is the nearest-neighbor code index, q_d is the quantized vector from that stage, and q is the final quantized output obtained by summing over all L stages. Euclidean distance is used for nearest-neighbor lookup. During training, the codebooks are updated via exponential moving average (EMA) rather than gradient descent. Cluster sizes and assigned embedding sums are maintained and synchronized across distributed nodes via all-reduce operations. Unused codes are reinitialized during training to improve codebook utilization and prevent collapse. The mask decoder reconstructs the generated discrete mask tokens into a segmentation mask. After the MLLM predicts a sequence of mask tokens, the tokenizer maps them back to the corresponding codebook entries, recovers the compact quantized embedding, and expands it into multiple mask embeddings. These mask embeddings are treated as sparse prompt token and fed, together with image embeddings, into the mask decoder–which follows SAM2's design and uses a two-layer two-way transformer– to produce the final binary mask. In this way, the mask tokenizer establishes a closed loop: between raster segmentation masks and autoregressive mask-token sequences. Segmentation masks are encoded into discrete tokens for language-model training and decoded back into pixel-level masks during inference. Notably, our mask tokenizer is pre-trained independently from the MLLM and can therefore be readily adopted by any MLLM for tasks requiring dense pixel understanding.

4.3 In-context learning

We construct visual-prompted in-context instruction data by reorganizing dense object detection annotations. For each image I, we first group object instances by category. Let

$$\mathcal{B}_c = \{b_{c,j}\}_{j=1}^{N_c} \tag{11}$$

denote the complete set of bounding boxes belonging to category c, where N_c is the number of corresponding instances. We sample a subset $\mathcal{P}_c \subseteq \mathcal{B}_c$ as visual prompts, while the supervision target remains the full box set $\mathcal{B}_c$. The in-context detection task is thus formulated as:

$$\hat{\mathcal{B}}_c = MLLM\left(I, \mathcal{P}_c\right), \tag{12}$$

where $\hat{\mathcal{B}}_c$ denotes the predicted bounding boxes of the category specified by the visual prompts. The model autoregressively generates the complete set of target boxes using the same coordinate tokens as our vector tokenizer, remaining compatible with the next-token prediction objective of the base MLLM. For images containing multiple categories, we organize the instructions in a multi-turn dialogue format, where each turn focuses on one category. For cross-image prompting, we select exemplar boxes from a support image I_s and use them to identify corresponding objects in a different query image.

4.4 Reinforcement learning with verifiable rewards (RLVR)

After supervised fine-tuning, the model can already follow multimodal prompts and emit structured geometric tokens. However, maximum-likelihood training treats each token independently and cannot directly optimize the geometric quality of the decoded output. Small token-level errors can propagate into large spatial deviations, invalid boxes, unmatched instances, or over-complex polygons. We therefore align the model via RLVR, where the reward is computed from the parsed geometric prediction against the reference annotation.

Rollout pipeline. Given an input remote sensing image and prompt x, the policy model π_θ samples a group of K candidate responses $\{y_i\}_{i=1}^{K}$ via the rollout engine. Each response is parsed into a task-specific geometric structure and evaluated by a verifiable reward function $R(y_i, y^*, m)$, where y^* is the ground-truth annotation and m encodes task metadata such as output type and coordinate resolution.

Reward system. The reward function is task-specific and supports bounding boxes, polygons, and mask tokens. For all structured geometric outputs, predictions are expected to follow a JSON-array format where each geometry record is paired with its category phrase. If JSON parsing fails, the system applies fallback parsing to recover geometric tokens from raw text. Successfully recovered fallback outputs incur a format penalty of 0.8, while unrecoverable or malformed predictions receive zero reward. This design makes format validity and parseability part of the alignment signal. For bounding-box outputs, both predictions and references are grouped by semantic phrase. For a phrase p, let $\mathcal{B}_p$ and $\mathcal{B}_p^*$ denote the predicted and reference

boxes. We compute a bidirectional best-match IoU:

$$R_{\text{box}}(p) = \frac{1}{2}\left(\frac{1}{|\mathcal{B}_p|}\sum_{b\in\mathcal{B}_p}\max_{b^*\in\mathcal{B}_p^*}\text{IoU}(b,b^*) + \frac{1}{|\mathcal{B}_p^*|}\sum_{b^*\in\mathcal{B}_p^*}\max_{b\in\mathcal{B}_p}\text{IoU}(b,b^*)\right). \tag{13}$$

The final box reward is averaged across phrases. This bidirectional formulation rewards accurate localization while penalizing both missing ground-truth objects and redundant predictions. For polygon outputs, each predicted polygon token sequence is parsed into vertices and rasterized into a binary mask. The primary term is a phrase-level bidirectional polygon mask IoU, computed analogously to the box reward. To encourage compact and editable vector outputs, we further introduce a node-count reward. Predicted and reference polygons are paired using Hungarian matching on the IoU matrix, and each matched pair receives a vertex-efficiency score that penalizes predictions with more vertices than their matched reference. The final polygon reward is

$$R_{\text{poly}} = 0.75R_{\text{IoU}} + 0.25R_{\text{node}}. \tag{14}$$

This reward directly encourages region agreement while discouraging over-fragmented polygon boundaries. For segmentation masks, outputs are represented as ordered sequences of special tokens. Each predicted mask is compared with the reference sequence using prefix consistency: the pairwise score is the ratio of consecutively matching tokens from the start of the sequence to the total mask-token length. Phrase-level and final mask rewards are then computed with the same bidirectional best-match aggregation used for bounding boxes. This provides a verifiable alignment signal for mask-token generation without requiring a learned reward model. For VQA tasks, we apply rule-based rewards only for verifiable answer types, including binary (yes/no) questions and multiple-choice questions with explicit options. VQA predictions are free-form text. The system first attempts to extract an answer field from a JSON-style response, and otherwise normalizes the raw text. For binary questions, variants such as "yes/no" and "true/false" are mapped to Boolean labels. For multiple-choice questions, the response is parsed into a legal option label (e.g., $A/B/C/D$). The reward is 1 for exact matches, and 0 otherwise. Unparseable or unsupported open-ended answers receive zero reward.

Hard sample mining. To improve the usefulness of RL samples, we estimate candidate difficulty before training by generating multiple rollouts and scoring them with the same reward functions. Samples with uniformly zero or uniformly perfect rewards are removed, since they provide little relative learning signal for group relative policy optimization (GRPO) [33, 34]. The remaining samples are binned by reward-based difficulty and sampled to maintain a balanced mix of easy, medium, and hard examples. Overall, perceptual RLVR translates geometric correctness, phrase grounding, format validity, and vector compactness into direct optimization signals, pushing the model to move beyond token-level imitation toward mapping-grade geometric generation.

Optimization of the policy model. We optimize the model with GRPO. For each prompt, rewards within the sampled group are normalized to produce relative

advantages:

$$A_i = \frac{R_i - \text{mean}(\{R_j\}_{j=1}^K)}{\text{std}(\{R_j\}_{j=1}^K) + \epsilon}. \tag{15}$$

The policy model is then updated using a clipped policy-gradient objective, with a Kullback–Leibler (KL) regularization term that constrains it to remain close to the reference model. In our implementation, KL is applied in the actor loss rather than being added to the scalar reward. This preserves the interpretability of the verifiable reward while stabilizing optimization.

4.5 Training pipeline

This subsection summarizes the training pipeline for VectorLLM++. We first construct the training data using our vector-instruction data engine together with publicly available datasets. The mask tokenizer is pre-trained separately, after which both the mask and vector tokenizers are used to pre-tokenize the data for the three MLLM training stages. The full model is then optimized through large-scale pre-training, supervised fine-tuning (SFT), and reinforcement learning post-training. The resulting model is a remote sensing MLLM that supports seven distinct tasks under a unified next-token prediction framework.

Mask tokenizer training. We train RSM-Tok independently before MLLM training. Its training data are constructed by mixing curated remote sensing segmentation masks (the same for training dense segmentation task) with SA-1B masks [57] at a 1:1 ratio, so that the tokenizer learns both remote sensing object masks and general binary-mask geometry. Given a binary mask M, the mask tokenizer encodes it into residual-quantized mask tokens and decodes them back into a reconstructed mask $\widehat{M}$. Following SAMTok [56], we optimize the tokenizer with reconstruction and codebook losses:

$$\begin{aligned}\mathcal{L}_{\text{mtok}} = \mathcal{L}_{\text{BCE}}(\widehat{M}, M) + \lambda_{\text{dice}}\mathcal{L}_{\text{Dice}}(\widehat{M}, M) \\ + \lambda_{\text{cb}} \sum_{d=1}^{L} \|\text{sg}(r_{d-1}) - q_d\|_2^2 + \lambda_{\text{com}} \sum_{d=1}^{L} \|r_{d-1} - \text{sg}(q_d)\|_2^2,\end{aligned} \tag{16}$$

where r_{d-1} is the residual input to the d-th quantization stage, q_d is the selected codebook vector, and $\text{sg}(\cdot)$ denotes stop-gradient. The first two terms are reconstruction losses: $\mathcal{L}_{\text{BCE}}$ provides pixel-wise binary supervision, while $\mathcal{L}_{\text{Dice}}$ improves region-level overlap and alleviates foreground-background imbalance. The third term, the codebook loss, drives the selected codebook embeddings toward the encoder residuals. The fourth term, the commitment loss, encourages the encoder output to stay close to the selected codes and stabilizes quantization. After training, the mask tokenizer– comprising the mask encoder, quantizer, and decoder–is frozen and serves as the mask serialization interface for VectorLLM++.

Offline tokenization of training data. We then pre-tokenize all geometric annotations. Bounding boxes and polygons are normalized to the image extent and serialized by the vector tokenizer into discrete coordinate tokens. Segmentation masks are encoded by the frozen mask tokenizer into ordered mask tokens. Language answers,

category names, referring expressions, and VQA responses remain standard text tokens. Therefore, each example is converted into a multimodal instruction tuple (I, q, s), where I denotes the image (or image pair), q is the instruction with optional visual prompts, and s is the target token sequence. All subsequent MLLM training uses the same next-token prediction objective:

$$\mathcal{L}_{\text{ntp}} = -\sum_{t=1}^{T} \log p_\theta(s_t \mid s_{<t}, I, q). \tag{17}$$

Large-scale pre-training. Starting from the Qwen3-VL backbone equipped with the vector tokenizer and mask tokenizer, we first pre-train VectorLLM++ on large-scale detection and segmentation data. This stage uses 62.9M image-level samples summarized in Fig. 1b, comprising 11.0M detection samples without semantic labels, 21.2M detection samples with semantic labels, 11.0M segmentation samples without semantic labels, and 19.7M segmentation samples with semantic labels. This stage adapts the base MLLM to remote sensing visual grounding and dense geometric generation before introducing the full task mixture.

Supervised fine-tuning. After pre-training, we fine-tune the model on a high-quality instruction mixture covering seven capabilities: dense detection, referring detection, in-context detection, dense segmentation, dense polygon extraction, change detection, and VQA. The dense polygon extraction data include the GeoPolygon train split generated by our data engine, containing 1,213,855 polygon dialogues and 1,834 target phrases. Together with public datasets, the SFT mixture contains 7,193,499 samples: 544,575 detection, 163,903 change-detection, 3,723,132 segmentation, 1,993,513 polygon-extraction, and 768,376 VQA samples. This stage teaches the model the final instruction formats and aligns each task with its corresponding output grammar.

Post-training with GRPO. Finally, we initialize from the SFT checkpoint and apply the RLVR procedure described above. We generate candidate rollouts, compute task-specific verifiable rewards, discard samples that consistently yield zero or perfect rewards, and keep the remaining hard samples for GRPO training. The RL stage uses 47,701 mined samples, including 9,800 samples each for detection, change detection, segmentation, and polygon extraction, and 8,501 samples for VQA. This stage directly optimizes output validity and geometric quality, finally yielding a single VectorLLM++ model that supports all seven remote sensing perception and reasoning tasks.

References

[1] Zhu, X.X., Tuia, D., Mou, L., Xia, G.-S., Zhang, L., Xu, F., Fraundorfer, F.: Deep learning in remote sensing: A comprehensive review and list of resources. IEEE geoscience and remote sensing magazine **5**(4), 8–36 (2017)

[2] Zhang, X., Zhang, T., Wang, G., Zhu, P., Tang, X., Jia, X., Jiao, L.: Remote sensing object detection meets deep learning: A metareview of challenges and advances. IEEE Geoscience and Remote Sensing Magazine **11**(4), 8–44 (2023)

[3] Li, K., Wan, G., Cheng, G., Meng, L., Han, J.: Object detection in optical remote sensing images: A survey and a new benchmark. ISPRS journal of photogrammetry and remote sensing **159**, 296–307 (2020)

[4] Li, Z., Wegner, J.D., Lucchi, A.: Topological map extraction from overhead images. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 1715–1724 (2019)

[5] Xu, B., Xu, J., Xue, N., Xia, G.: Hisup: Accurate polygonal mapping of buildings in satellite imagery with hierarchical supervision. ISPRS Journal of Photogrammetry and Remote Sensing **198**, 284–296 (2023)

[6] Wei, S., Zhang, T., Ji, S., Luo, M., Gong, J.: Buildmapper: A fully learnable framework for vectorized building contour extraction. ISPRS journal of photogrammetry and remote sensing **197**, 87–104 (2023)

[7] Kruitwagen, L., Story, K., Friedrich, J., Byers, L., Skillman, S., Hepburn, C.: A global inventory of photovoltaic solar energy generating units. Nature **598**(7882), 604–610 (2021)

[8] Paolo, F.S., Kroodsma, D., Raynor, J., Hochberg, T., Davis, P., Cleary, J., Marsaglia, L., Orofino, S., Thomas, C., Halpin, P.: Satellite mapping reveals extensive industrial activity at sea. Nature **625**(7993), 85–91 (2024)

[9] Li, Z., Li, L., Hu, T., Cheng, M., He, W., Qiu, T., Zhang, L., Zhang, H.: Satellite mapping of every building's function in urban china reveals deep built environment disparities. Nature Communications **17**(1), 2827 (2026)

[10] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., *et al.*: Language models are few-shot learners. Advances in neural information processing systems **33**, 1877–1901 (2020)

[11] Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F.L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al.: Gpt-4 technical report. arXiv preprint arXiv:2303.08774 (2023)

[12] Alayrac, J.-B., Donahue, J., Luc, P., Miech, A., Barr, I., Hasson, Y., Lenc, K., Mensch, A., Millican, K., Reynolds, M., *et al.*: Flamingo: a visual language model for few-shot learning. Advances in neural information processing systems **35**, 23716–23736 (2022)

[13] Liu, H., Li, C., Wu, Q., Lee, Y.J.: Visual instruction tuning. Advances in neural information processing systems **36**, 34892–34916 (2023)

[14] Bai, S., Cai, Y., Chen, R., Chen, K., Chen, X., Cheng, Z., Deng, L., Ding, W., Gao, C., Ge, C., et al.: Qwen3-vl technical report. arXiv preprint

arXiv:2511.21631 (2025)

[15] Hu, Y., Yuan, J., Wen, C., Lu, X., Liu, Y., Li, X.: Rsgpt: A remote sensing vision language model and benchmark. ISPRS Journal of Photogrammetry and Remote Sensing **224**, 272–286 (2025)

[16] Zhan, Y., Xiong, Z., Yuan, Y.: Skyeyegpt: Unifying remote sensing vision-language tasks via instruction tuning with large language model. ISPRS Journal of Photogrammetry and Remote Sensing **221**, 64–77 (2025)

[17] Muhtar, D., Li, Z., Gu, F., Zhang, X., Xiao, P.: Lhrs-bot: Empowering remote sensing with vgi-enhanced large multimodal language model. In: European Conference on Computer Vision, pp. 440–457 (2024). Springer

[18] Kuckreja, K., Danish, M.S., Naseer, M., Das, A., Khan, S., Khan, F.S.: Geochat: Grounded large vision-language model for remote sensing. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 27831–27840 (2024)

[19] Zhou, Y., Lan, M., Li, X., Feng, L., Ke, Y., Jiang, X., Li, Q., Yang, X., Zhang, W.: Geoground: A unified large vision-language model for remote sensing visual grounding. arXiv preprint arXiv:2411.11904 (2024)

[20] Shabbir, A., Zumri, M., Bennamoun, M., Khan, F.S., Khan, S.: Geopixel: Pixel grounding large multimodal model in remote sensing. In: Forty-second International Conference on Machine Learning. ICML'25 (2025)

[21] Wu, K., Zhang, Y., Ru, L., Dang, B., Lao, J., Yu, L., Luo, J., Zhu, Z., Sun, Y., Zhang, J., *et al.*: A semantic-enhanced multi-modal remote sensing foundation model for earth observation. Nature Machine Intelligence **7**(8), 1235–1249 (2025)

[22] Zhu, X.X., Xiong, Z., Wang, Y., Stewart, A.J., Heidler, K., Wang, Y., Yuan, Z., Dujardin, T., Xu, Q., Shi, Y.: On the foundations of earth foundation models. Communications Earth & Environment **7**(1), 103 (2026)

[23] EO, E.O.: Towards responsible geospatial foundation models. Nat. Mach. Intell **7**, 1189 (2025)

[24] Zhang, T., Wei, S., Chen, S., Yu, W., Luo, M., Ji, S.: Vectorllm: Human-like extraction of structured building contours via multimodal llms. ISPRS Journal of Photogrammetry and Remote Sensing **233**, 55–68 (2026)

[25] Van Den Oord, A., Vinyals, O., et al.: Neural discrete representation learning. Advances in neural information processing systems **30** (2017)

[26] Chen, T., Saxena, S., Li, L., Fleet, D.J., Hinton, G.: Pix2seq: A language modeling framework for object detection. In: International Conference on Learning

Representations (2022)

[27] Chen, T., Saxena, S., Li, L., Lin, T.-Y., Fleet, D.J., Hinton, G.E.: A unified sequence interface for vision tasks. Advances in Neural Information Processing Systems **35**, 31333–31346 (2022)

[28] Lai, X., Tian, Z., Chen, Y., Li, Y., Yuan, Y., Liu, S., Jia, J.: Lisa: Reasoning segmentation via large language model. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9579–9589 (2024)

[29] Rasheed, H., Maaz, M., Shaji, S., Shaker, A., Khan, S., Cholakkal, H., Anwer, R.M., Xing, E., Yang, M.-H., Khan, F.S.: Glamm: Pixel grounding large multimodal model. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 13009–13018 (2024)

[30] Yuan, H., Li, X., Zhang, T., Sun, Y., Huang, Z., Xu, S., Ji, S., Tong, Y., Qi, L., Feng, J., et al.: Sa2va: Marrying sam2 with llava for dense grounded understanding of images and videos. arXiv preprint arXiv:2501.04001 (2025)

[31] Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., *et al.*: Training language models to follow instructions with human feedback. Advances in neural information processing systems **35**, 27730–27744 (2022)

[32] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O.: Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347 (2017)

[33] Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Bi, X., Zhang, H., Zhang, M., Li, Y., Wu, Y., et al.: Deepseekmath: Pushing the limits of mathematical reasoning in open language models. arXiv preprint arXiv:2402.03300 (2024)

[34] Guo, D., Yang, D., Zhang, H., *et al.*: Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. Nature **645**, 633–638 (2025) https://doi.org/10.1038/s41586-025-09422-z

[35] Lobry, S., Marcos, D., Murray, J., Tuia, D.: Rsvqa: Visual question answering for remote sensing data. IEEE Transactions on Geoscience and Remote Sensing **58**(12), 8555–8566 (2020)

[36] Minderer, M., Gritsenko, A., Stone, A., Neumann, M., Weissenborn, D., Dosovitskiy, A., Mahendran, A., Arnab, A., Dehghani, M., Shen, Z., *et al.*: Simple open-vocabulary object detection. In: European Conference on Computer Vision, pp. 728–755 (2022). Springer

[37] Zhou, X., Girdhar, R., Joulin, A., Krähenbühl, P., Misra, I.: Detecting twenty-thousand classes using image-level supervision. In: European Conference on Computer Vision, pp. 350–368 (2022). Springer

[38] Huang, X., Huang, Y.-J., Zhang, Y., Tian, W., Feng, R., Zhang, Y., Xie, Y., Li, Y., Zhang, L.: Open-set image tagging with multi-grained text supervision. In: Proceedings of the 33rd ACM International Conference on Multimedia, pp. 4117–4126 (2025)

[39] Carion, N., Gustafson, L., Hu, Y.-T., Debnath, S., Hu, R., Suris, D., Ryali, C., Alwala, K.V., Khedr, H., Huang, A., *et al.*: Sam 3: Segment anything with concepts. In: International Conference on Learning Representations, vol. 2026, pp. 138846–138923 (2026)

[40] Liu, Z., Yuan, L., Weng, L., Yang, Y.: A high resolution optical satellite image dataset for ship recognition and some new baselines. In: International Conference on Pattern Recognition Applications and Methods, vol. 2, pp. 324–331 (2017). SciTePress

[41] Ji, S., Wei, S., Lu, M.: Fully convolutional networks for multisource building extraction from an open aerial and satellite imagery data set. IEEE Transactions on geoscience and remote sensing **57**(1), 574–586 (2018)

[42] Luo, M., Ji, S., Wei, S.: A diverse large-scale building dataset and a novel plug-and-play domain generalization method for building extraction. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing **16**, 4122–4138 (2023)

[43] Li, Q., Ma, S., Luo, J., Yu, Y., Zhou, Y., Wang, F., Lu, X., Wang, X., He, X., Chen, Y., *et al.*: Co-training vision-language models for remote sensing multi-task learning. Remote Sensing **18**(2), 222 (2026)

[44] Jiang, Q., Huo, J., Chen, X., Xiong, Y., Zeng, Z., Chen, Y., Ren, T., Yu, J., Zhang, L.: Detect anything via next point prediction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 25472–25483 (2026)

[45] Yao, K., Xu, N., Yang, R., Xu, Y., Gao, Z., Kitrungrotsakul, T., Ren, Y., Zhang, P., Wang, J., Wei, N., et al.: Falcon: A remote sensing vision-language foundation model (technical report). arXiv preprint arXiv:2503.11070 (2025)

[46] Ni, S., Wang, D., Chen, H., Guo, H., Zhang, N., Zhang, J.: Unigeoseg: Towards unified open-world segmentation for geospatial scenes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 34865–34876 (2026)

[47] Shi, Q., He, D., Liu, Z., Liu, X., Xue, J.: Globe230k: A benchmark dense-pixel annotation dataset for global land cover mapping. Journal of Remote Sensing **3**, 0078 (2023)

[48] Wang, D., Zhang, J., Du, B., Xu, M., Liu, L., Tao, D., Zhang, L.: Samrs: Scaling-up remote sensing segmentation dataset with segment anything model. Advances in Neural Information Processing Systems **36**, 8815–8827 (2023)

[49] Wang, J., Zheng, Z., Ma, A., Lu, X., Zhong, Y.: Loveda: A remote sensing land-cover dataset for domain adaptive semantic segmentation. In: Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks, vol. 1 (2021)

[50] Zhan, Y., Xiong, Z., Yuan, Y.: Rsvg: Exploring data and models for visual grounding on remote sensing data. IEEE transactions on geoscience and remote sensing **61**, 1–13 (2023)

[51] Shi, Q., Liu, M., Li, S., Liu, X., Wang, F., Zhang, L.: A deeply supervised attention metric-based network and an open aerial image dataset for remote sensing change detection. IEEE transactions on geoscience and remote sensing **60**, 1–16 (2021)

[52] Lebedev, M., Vizilter, Y.V., Vygolov, O., Knyaz, V.A., Rubis, A.Y.: Change detection in remote sensing images using conditional adversarial networks. The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences **42**, 565–571 (2018)

[53] Zhang, Z., Jiao, L., Li, L., Liu, X., Chen, P., Liu, F., Li, Y., Guo, Z.: A spatial hierarchical reasoning network for remote sensing visual question answering. IEEE Transactions on Geoscience and Remote Sensing **61**, 1–15 (2023)

[54] Soni, S., Dudhane, A., Debary, H., Fiaz, M., Munir, M.A., Danish, M.S., Fraccaro, P., Watson, C.D., Klein, L.J., Khan, F.S., *et al.*: Earthdial: Turning multi-sensory earth observations to interactive dialogues. In: Proceedings of the Computer Vision and Pattern Recognition Conference, pp. 14303–14313 (2025)

[55] Ravi, N., Gabeur, V., Hu, Y.-T., Hu, R., Ryali, C., Ma, T., Khedr, H., Rädle, R., Rolland, C., Gustafson, L., *et al.*: Sam 2: Segment anything in images and videos. In: International Conference on Learning Representations, vol. 2025, pp. 28085–28128 (2025)

[56] Zhou, Y., Zhang, T., Gong, D., Wu, Y., Tian, Y., Wang, H., Yuan, H., Wang, J., Qi, L., Fei, H., *et al.*: Samtok: Representing any mask with two words. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 37852–37863 (2026)

[57] Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A.C., Lo, W.-Y., *et al.*: Segment anything. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 4015–4026 (2023)

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- NCVectorIlmv2finalsupp.pdf