

YOUR MODEL ALREADY KNOWS DON'T TEACH IT, LEARN TO ASK IT



SOFT PROMPTING FOR FEW-SHOT ADAPTATION OF VISION-LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Few-shot adaptation of vision-language models (VLMs) is difficult for specialized aerial, industrial, medical, and other imagery when only ten annotated images are available. Discrete prompt optimizers such as GEPA and DetPO keep the model frozen but search text without numerical gradients; low-rank adapters use gradient descent but learn millions of task-specific parameters. We revisit soft prompting, which optimizes a few continuous input tokens by gradient descent against a frozen backbone, and find that two defaults inherited from NLP hold it back: prefix placement and semantic initialization. Placing the tokens at the cross-modal boundary, between image and text, and initializing them from a semantically empty space token unlock its performance. On Roboflow20-VL, the resulting prompt uses 7,168 trainable parameters on average, outperforms GEPA and DetPO under a shared multi-class protocol, and matches the best LoRA configuration at 14.2 mAP@50:95 with roughly 24,000× fewer trainable parameters. The learned tokens behave like prompts rather than weights: they transfer to a newer model version without retraining, and SOFT2HARD verbalizes them into an editable instruction that matches DetPO. The placement principle also carries beyond detection: on RoboCasa, a prompt that reaches the action expert of a frozen robot policy raises two near-floor tasks from 5.0% to 23.3% and 31.7% success. Our results suggest that modern VLMs already encode much of what specialized domains require; the task is not to teach them, but to learn how to ask.

1 INTRODUCTION

Vision-language models (VLMs) trained on web-scale data recognize common objects, yet often fail on specialized imagery such as aerial surveys, industrial inspection, and medical scans (Radford et al., 2021; Bai et al., 2025; Robicheaux et al., 2025). When only ten labeled images are available, the central question is how much task-specific machinery should be learned.

Existing approaches occupy two ends of a spectrum. Discrete prompt optimization, including GEPA (Agrawal et al., 2026), DetPO (Gare et al., 2026), and TextGrad (Yuksekgonul et al., 2024), keeps the VLM frozen and searches over human-readable tokens, but does not apply numerical gradient descent to continuous prompt embeddings. LoRA (Hu et al., 2022) instead freezes the pretrained matrices and learns low-rank factors whose updates are active with the adapter; it supports gradient descent but introduces a task-specific parameter block at many transformer layers. Soft prompting combines the relevant benefits: like LoRA, it optimizes continuous parameters by numerical gradient descent; like prompting, it leaves the pretrained backbone frozen and attaches task-specific state only at the input.

Soft prompting is not automatically effective in a multimodal autoregressive sequence, and we find that two defaults inherited from NLP (Lester et al., 2021; Li & Liang, 2021) hold it back: learned vectors are prepended to the sequence and seeded with task semantics. A VLM sequence, however, has distinct visual and textual spans and therefore several possible boundaries. Placing the tokens at the cross-modal boundary, between the visual patches and the text query, wins our six-way ordering

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.

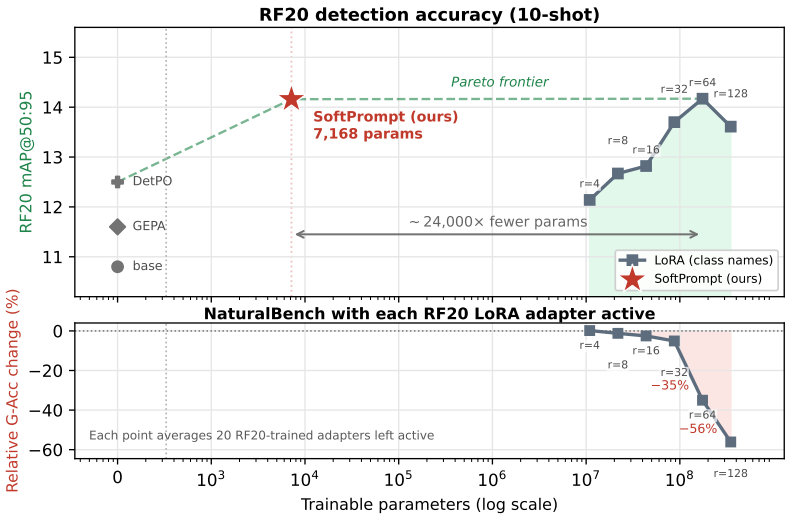

Figure 1: **SOFTPROMPT matches the best LORA rank with $\sim$24,000$\times$ fewer trainable parameters and zero forgetting.** *Top:* RF20 10-shot mAP@50:95 against trainable parameters (log scale). Training-free prompt optimizers (GEPA, DetPO) stay at or below 12.5. Sweeping LORA rank from 4 to 128 (squares) peaks at 14.2 at r=64 (174.6M parameters) and then declines, so more adapter capacity does not keep helping. SOFTPROMPT (star) reaches the same 14.2 with 7,168 parameters on average. *Bottom:* the cost that detection-only evaluation misses. Relative change in NaturalBench group accuracy (G-Acc, a question group counts only if all its answers are correct), averaged over 20 RF20-trained LORA adapters per rank. The two panels move in opposite directions: the ranks that help detection most damage the generalist most (-35% at r=64, -56% at r=128). SOFTPROMPT never changes the backbone, so every other task is served by the unchanged model.

ablation (10.5 average mAP vs. 9.6 for the NLP prefix, with a $\sim$6$\times$ swing on one dataset), and initializing them from a space token, so that the query's meaning is unchanged at the start, beats semantic and random starts (Section 5). With these choices, SOFTPROMPT uses one to three learned tokens per domain and reaches 14.2 mAP on Roboflow20-VL, matching the best configuration in a six-rank LORA sweep and exceeding multi-class DetPO by 1.7 mAP (Figure 1).

The learned tokens also keep the practical properties of text prompts; in this sense they behave like prompts rather than weights. Loaded as-is into the newer Qwen3.5-9B, they still improve over that model's own baseline (Section 4.4). Our SOFT2HARD procedure asks the VLM to verbalize the task information elicited by its learned soft prompt and produces an editable natural-language instruction competitive with dedicated prompt search (Section 4.3); the text is an inspectable artifact, not a faithful causal explanation of each embedding dimension. Finally, the placement principle carries beyond detection: on a frozen robot policy, the soft prompt helps only once it reaches the module that produces the output (Section 4.6).

Contributions. First, we identify the two choices that make soft prompting competitive for multimodal detection, cross-modal placement and meaning-preserving initialization, and combine them with short length sweeps into a practical few-shot recipe. Second, in a controlled Roboflow20-VL comparison against prompt search, a six-rank LORA sweep, and a specialist detector, we show that the resulting prompt reaches the LORA frontier with a small fraction of its trainable parameters. Finally, we show that the learned tokens behave like prompts: they transfer across model versions, can be verbalized by SOFT2HARD into editable text, and follow the same placement principle on a robot policy.

2 RELATED WORK

VLM detection. Generative VLMs such as Qwen3-VL (Bai et al., 2025), LLaVA (Liu et al., 2023), and InstructBLIP (Dai et al., 2023) emit detections as structured text. Specialist open-vocabulary

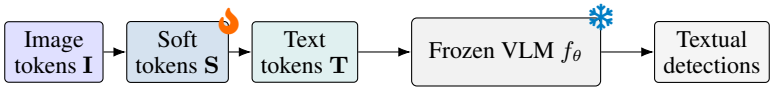

Figure 2: **SOFTPROMPT inserts L learned tokens at the cross-modal boundary (IST), after every image patch and before every query token.** Placed there, the tokens can read the whole image and steer every query token. The VLM stays frozen (❄): gradients flow back through all its layers, but only the $L \times d$ embeddings in $\mathbf{S}$ are trained (🔥).

detectors such as Grounding DINO (Liu et al., 2024) and OWL-ViT (Minderer et al., 2022) align regions with text and remain stronger in raw detection accuracy; we include them to delimit our claim about adapting a general-purpose VLM.

Prompt tuning and PEFT. Prompt tuning (Lester et al., 2021), prefix tuning (Li & Liang, 2021), VPT (Jia et al., 2022), CoOp/CoCoOp (Zhou et al., 2022a;b), and MaPLe (Khattak et al., 2023) learn continuous prompts for frozen models. Like these methods, we train only continuous prompt vectors, but ours is shallow rather than deep prompt tuning. Deep variants such as VPT-Deep and P-Tuning v2 (Liu et al., 2022) inject a fresh learnable prompt at every transformer layer; our soft tokens enter the sequence once, at the input, as in VPT-Shallow, and every layer's weights stay frozen. Prior multimodal prompt tuning also mostly targets contrastive CLIP-style encoders; in a generative VLM, the prompt instead sits inside one autoregressive sequence of image and text tokens, so its position relative to the two modalities becomes a first-class design variable, which we show is decisive (Section 5.1). LoRA (Hu et al., 2022) and adapters (Houlsby et al., 2019) are the weight-space alternatives we compare against. SPoT (Vu et al., 2022) studied prompt transfer for text-only models; Section 4.4 gives a multimodal counterpart.

Discrete prompt optimization. GEPA (Agrawal et al., 2026) evolves prompts, TextGrad (Yuksekgonul et al., 2024) propagates language critiques, AutoPrompt (Shin et al., 2020) performs gradient-guided token search, and DetPO (Gare et al., 2026) specializes discrete search to detection. These methods preserve readable tokens but search a discrete space; our method uses gradients in continuous embedding space and then recovers an editable hard prompt through SOFT2HARD. Additional work on distillation and transfer appears in Appendix D.

3 METHOD

3.1 SOFT TOKENS AT THE CROSS-MODAL BOUNDARY

A VLM f_θ encodes an image as N_I patch embeddings $\mathbf{I} \in \mathbb{R}^{N_I \times d}$ and a query as N_T token embeddings $\mathbf{T} \in \mathbb{R}^{N_T \times d}$, then emits detections as structured text. We add L trainable embeddings $\mathbf{S} \in \mathbb{R}^{L \times d}$. Image (I), soft (S), and text (T) spans admit six orderings: IST, TSI, ITS, TIS, SIT, and STI. Our default places the prompt between modalities,

$$\widetilde{\mathbf{X}} = [\mathbf{I}; \mathbf{S}; \mathbf{T}], \tag{1}$$

so the learned tokens follow every visual patch and precede every query token. SIT and STI transpose the NLP prefix, placing the learned state before either modality; ITS and TIS make it a suffix; IST and TSI place it at a modality boundary. In IST, the prompt is conditioned by every preceding visual patch while affecting every following query token, a plausible bottleneck for domain-specific visual evidence. Because the best ordering may depend on the backbone's serialization, we evaluate all six and fix IST only after the controlled comparison (Section 5.1). Figure 2 summarizes the design; Appendices F and J detail gradient flow and serving with a standard inference engine.

3.2 OPTIMIZATION AND INITIALIZATION

Given the 10-shot support set $\mathcal{D}$, training uses the backbone's next-token detection loss while freezing θ: for an image–query pair with target detection string $\mathbf{y} = (y_1, \ldots, y_M)$, we minimize over $\mathbf{S}$

$$\mathcal{L}_{\text{det}}(\mathbf{S}; \theta) = -\sum_{m=1}^{M} \log p_\theta(y_m \mid y_{<m}, \mathbf{I}, \mathbf{S}, \mathbf{T}), \qquad \theta \text{ fixed.} \tag{2}$$

The prompt gradient, $\nabla_{\mathbf{S}}\mathcal{L}_{\text{det}}$, nevertheless includes the Jacobian of every frozen transformer block between the input and the emitted detection tokens. "Frozen" therefore means that those blocks retain no updates, not that they disappear from forward or backward computation. The target is the same structured sequence used by the frozen detector, with class labels and normalized box coordinates generated autoregressively; no detection head, visual projection, or pretrained embedding row is updated.

This gives SOFTPROMPT and LORA different optimization geometry: LORA places trainable factors at many depths, whereas SOFTPROMPT has one input-side sink at the far end of the backward path, so its influence must propagate through every frozen layer (Appendix F). This motivates careful placement and a conservative initialization.

For Qwen3-VL-8B, $d = 4096$: $L = 1$, $L = 2$, and $L = 3$ contain $4{,}096$, $8{,}192$, and $12{,}288$ trainable parameters, respectively. The selected per-domain lengths average $L = 1.75$, or $7{,}168$ parameters. We initialize every row of $\mathbf{S}$ with the embedding of a space token, following the residual-learning principle (He et al., 2016): the safest default for a new module is to change nothing. A semantic seed such as a class name or the word `detect` changes the query before any evidence that the change helps, so optimization must preserve useful meaning, erase harmful meaning, and fit the task at once; a whitespace start leaves the query nearly unchanged and asks the optimizer to learn only the residual task information. We test this claim against Xavier-uniform, mean-vocabulary, class-seeded, and task-text initializations. The space tokens still occupy sequence positions, so we call the start meaning-preserving rather than function-preserving.

3.3 FROM CONTINUOUS TO EDITABLE PROMPTS

SOFT2HARD exposes the learned soft prompt through the model's own language. Directly asking what an isolated embedding means is ill-posed: the prompt was optimized only in the joint context of an image and detection query, and its nearest vocabulary token need not express its contextual effect. We therefore read out behavior rather than assign a word to each vector.

The procedure has three stages. First, with the learned soft prompt active, the VLM describes the target objects in every support image before producing detections. Second, a text-only pass summarizes these descriptions into one global detection instruction. Third, that instruction replaces the soft tokens and is evaluated with the unmodified base VLM. The deployed artifact is therefore ordinary text: it can be inspected, edited, versioned, or passed to a model API that exposes no embedding interface.

As a control, the same pipeline is run without soft tokens, since descriptions of labeled support images can improve a prompt on their own. Only the presence of $\mathbf{S}$ during description generation differs, so the soft-minus-base difference isolates what the continuous prompt adds through the text. Appendix D gives the full procedure and analyses.

4 EXPERIMENTS

4.1 SETUP

Benchmark and models. Roboflow20-VL is a 20-domain subset of Roboflow-100-VL (Robicheaux et al., 2025) spanning aerial, document, flora and fauna, industrial, medical, sports, and other imagery. Each domain supplies ten support images and a held-out test split. Main results use Qwen3-VL-8B-Instruct and COCO mAP@50:95 on a 0–100 scale. The VLM receives one image and a class-name query, then emits all detections in one structured response. We parse class labels and normalized coordinates and score them with the same evaluator for every generalist method. Category columns average datasets within each of the benchmark's seven super-categories; "All" gives equal weight to each of the 20 domains. Appendix B reports a higher-baseline LVIS Rare 50 check, and Appendix I trains the recipe on the unrelated Gemma-4-12B-it family.

Baselines. HARDPROMPT queries the frozen VLM with class names or hand-written instructions. GEPA and DetPO optimize discrete prompts on the same frozen backbone. LORA sweeps ranks $4, 8, 16, 32, 64, 128$; it freezes pretrained matrices and trains low-rank factors on attention and MLP projections. Every rank uses the same 10-shot support images, optimizer budget, generation template,

Table 1: **SOFTPROMPT matches the best LORA configuration on Roboflow20-VL 10-shot with 7.2K instead of 174.6M trainable parameters.** Category-level mAP@50:95 (F. & F.: flora and fauna); All weights the 20 datasets equally; Params counts task-specific trainable parameters. **Bold**: best generalist result per column. The parity is aggregate: LORA leads on Aerial and Documents, SOFTPROMPT on Sports and Other. Medical stays near zero for every generalist while the fine-tuned specialist reaches 35.3, marking what ten examples cannot teach.

Method	Params	Aerial	Doc.	F. & F.	Ind.	Med.	Sports	Other	All
HARDPROMPT (class names)	0	6.4	7.5	25.0	8.2	0.0	8.6	9.1	10.8
HARDPROMPT + instructions	0	7.4	8.5	25.0	9.1	0.2	9.7	9.6	11.3
GEPA (Agrawal et al., 2026)	0	6.3	10.8	22.4	8.5	0.1	12.8	11.5	11.6
DetPO (Gare et al., 2026)	0	8.6	10.1	26.7	**9.9**	0.1	11.0	10.5	12.5
SOFTPROMPT (IST, per-domain L)	7.2K	9.0	14.6	27.7	8.7	0.0	**13.8**	**14.3**	**14.2**
LORA ($r = 64$)	174.6M	**20.2**	**19.9**	**28.6**	7.1	**0.7**	9.6	9.4	**14.2**
Grounding DINO (zero-shot)	0	31.2	5.0	34.3	13.0	0.4	5.7	16.9	17.3
Grounding DINO (10-shot fine-tune)	172M	40.6	32.0	44.9	41.0	35.3	30.8	26.8	35.7

and test evaluator; sweeping ranks keeps the parameter-efficiency comparison from resting on one arbitrary rank. Grounding DINO is a specialist reference, not a parameter-matched generalist. All main generalist rows use one multi-class query per image; DetPO's published single-class protocol reaches 17.5 mAP with VQA-score reranking, but requires one call per class and is not inference-cost-comparable (Appendix L).

Comparison scope. "Params" counts task-specific values learned by gradient descent and stored after training, not the shared frozen backbone; training-free prompt optimizers therefore have zero, although their search still consumes model calls.

Training and selection. SOFTPROMPT uses AdamW, learning rate 5×10^{-3}, cosine scheduling, and batch size 4. Thirteen domains use a five-epoch budget; seven use a 50-epoch budget with early stopping. LORA uses adapters on the decoder's attention and MLP projections, learning rate 5×10^{-5}, cosine scheduling, 0.2 warmup, effective batch size 2, and 70 optimizer steps. Both approaches use bf16 and seed 42 in the headline comparison. Appendix H gives the per-domain settings. The main SOFTPROMPT row fixes IST and chooses L per domain from the reported short length sweep, just as the LORA row takes the best rank of its sweep (Appendix G). **Disclosures.** (i) Results are single runs in a regime with an estimated 20–28% seed-to-seed spread (direct three-seed mean CV: 28%, Appendix R); differences below roughly one mAP point should be treated cautiously. (ii) The distillation and transfer analyses (Sections 4.3–4.4) start from the better of IST and TSI per domain rather than the fixed-IST headline prompt.

Ablation pool. Controlled ablations use nine fixed datasets covering every super-category: aerial-airport; actions; all-elements; defect-detection; dentalai; flir-camera-objects; lacrosse-object-detection, a multi-class sports dataset; new-defects-in-wood; and wb-prova. Dentalai is the floor control; the other eight are the *above-floor* datasets used in win counts. The pool is fixed across position, initialization, and length studies rather than chosen separately for each conclusion. We report dentalai even though its near-zero scores make wins unstable, and include it in numeric averages while excluding it from win counts. All full tables retain the frozen $L = 0$ class-name baseline so a learned soft prompt can be identified as harmful rather than merely worse than another learned configuration.

4.2 MAIN RF20 RESULTS

Table 1 shows the claimed middle ground: SOFTPROMPT uses numerical gradient descent and equals the best LORA rank at 14.2 mAP@50:95, but stores only 7.2K input parameters rather than a 174.6M per-layer adapter. Every other rank is lower, and adding capacity is not monotonic: rank 128 drops to 13.6 (Appendix K). The parity is aggregate rather than per-domain: LORA is stronger on Aerial and Documents, SOFTPROMPT on Sports and Other.

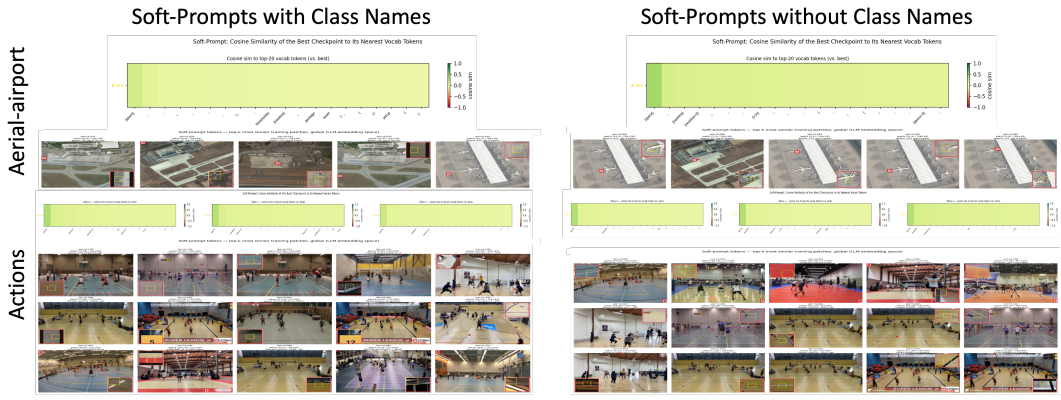

Figure 3: **What the tokens represent depends on what the query omits.** For trained soft tokens on an aerial and a sports dataset, the heatmaps show cosine similarity to the nearest vocabulary tokens and the image grids show the most similar training-image patches. When class names are withheld from the query, the tokens retrieve patches of the target object, so the prompt acts as a visual surrogate for the missing class name; when class names are present, as in our main protocol, they no longer do. In both settings the nearest word stays `[space]`, the initialization.

Against prompt baselines, hand-written instructions raise the frozen model from 10.8 to 11.3, GEPA reaches 11.6, and multi-class DetPO 12.5; gradient descent on a continuous prompt closes the further 1.7 mAP gap to the LoRA frontier.

The storage consequence is concrete. The mean SoftPrompt checkpoint is 29.6KB, whereas the accuracy-matching rank-64 LoRA adapter is 666.1MB (Appendix J). SoftPrompt also has zero forgetting by construction: the backbone weights never change, and like any prompt, a soft prompt affects only the queries it is attached to, so switching tasks is simply choosing a different input. LoRA adapters, by contrast, degrade the generalist: on NaturalBench VQA, the accuracy-matching rank reduces group accuracy by 35% relative and rank 128 by 56% (Figure 1, bottom; Appendix C).

The specialist detector defines the boundary: fine-tuned Grounding DINO reaches 35.7 mAP, and on Medical, where generalist prompts and adapters top out at 1.0, it reaches 35.3; ten examples can steer knowledge a general VLM already has, but not create it (qualitative examples in Appendix S).

Finding. 7.2K soft-prompt parameters reach the same 14.2 all-domain mAP as the best of six LoRA ranks (174.6M parameters) and exceed discrete prompt search by 1.7 mAP; the parity is aggregate, with LoRA stronger on Aerial and Documents and SoftPrompt on Sports and Other.

4.3 Editable verbalization

What the learned tokens encode depends on the query they complete (Figure 3). With class names withheld, their nearest visual patches depict the target object, so the prompt acts as a class surrogate; with class names present, as in our main protocol, they instead provide more general detection conditioning (Appendix D.1). Their meaning is therefore contextual, which is why Soft2Hard reads out behavior in context rather than naming individual vectors. Figure 4 shows the signal it turns into text.

Soft2Hard reaches 12.5 all-domain mAP as a plain-text prompt, matching DetPO and exceeding GEPA, while remaining below the soft prompt from which it is distilled. The control shows where this accuracy comes from. The same describe-and-summarize pipeline without soft tokens already reaches 12.2, so verbalization itself is a strong training-free baseline. The 12.5 versus 12.2 comparison, however, understates what the soft prompt contributes, because its effect is not a small uniform lift: it improves the distilled prompt on 10 of 20 domains, by $+1.9$ mAP on average, and lowers it on the other 10, by -1.4, and these partly cancel in the all-domain mean. The gains concentrate where the frozen model is weakest, in Aerial ($+4.1$) and Documents ($+2.6$), where the prompt-active descriptions add concrete visual distinctions such as the aircraft's relation to jet bridges in Figure 4; where the frozen model already describes a domain well, summarization can discard useful specificity.

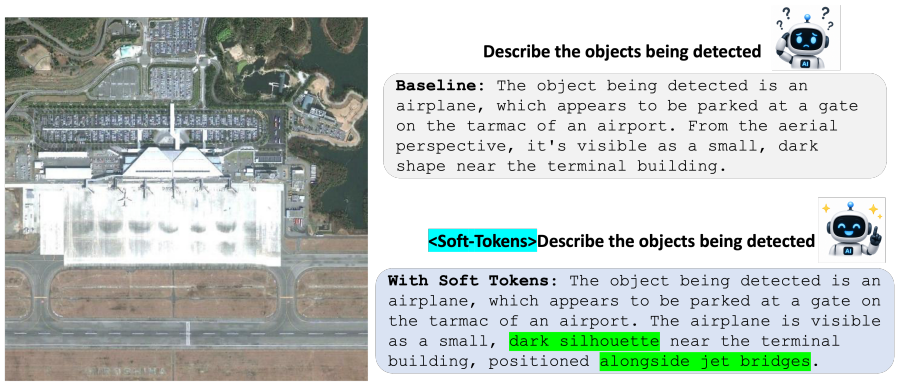

Figure 4: **The learned soft prompt elicits a more specific description.** Asked to describe the target in the same aerial image, the frozen model (top) sees only a "small, dark shape near the terminal"; with the learned prompt active (bottom), it adds the cues highlighted in green, a "dark silhouette" "alongside jet bridges". SOFT2HARD collects descriptions like these over the support set and summarizes them into an editable instruction for the base model, with no soft tokens needed.

Editability therefore has a price, retaining most but not all of the source prompt's accuracy, which is worth paying when a deployment values reviewability or text-only interfaces over the last points of mAP. A reviewer can inspect and edit the instruction and rerun the evaluation, but the sentence describes behavior, not the semantics of individual prompt vectors. Full results in Appendix D.

Finding. Verbalized into plain text, the learned soft prompt reaches 12.5 mAP, matching DetPO. The small gap to the no-soft-token recipe (12.2) hides a split rather than a uniform lift: the soft prompt helps 10 of 20 domains (+1.9 mAP on average, up to +4.1 on Aerial) and hurts the other 10 (−1.4).

4.4 TRANSFER ACROSS MODEL VERSIONS

A soft prompt is an input-side object rather than a set of changes tied to particular transformer modules. This creates a simple, if demanding, transfer test: load prompts trained on Qwen3-VL-8B directly into Qwen3.5-9B without a learned projection or any target-model training. The dimensions are compatible, but the embedding geometries need not be.

Across the 20 RF20 domains, direct transfer raises the 9B model from 9.9 to 10.7 mAP. It helps 13 domains by +2.9 mAP on average and hurts seven by −3.1, so the modest +0.8 aggregate is a mixture of meaningful gains and losses rather than a uniformly small effect. The transferred prompt also remains well below its score on the source model. Thus the embeddings carry some model-independent domain signal, but much of their utility remains coupled to the source model's geometry. The sign is domain-dependent even between successive releases of one family: transfer helps most on Sports (+5.3) and Documents (+3.3) but loses 2.9 points on Industrial (Appendix N). We compare against the target model's own unprompted score because the 9B backbone does not preserve the 8B model's per-domain strengths; with no projection or target-model gradient step, this is a deliberately minimal transfer baseline. Text from SOFT2HARD is the more portable artifact, since it can cross an API boundary, but raw embeddings are the stronger transfer channel when hidden dimensions match: they usually outperform their own distilled descriptions on the target model (Appendix O).

Finding. Loaded without retraining into Qwen3.5-9B, the soft prompts lift it from 9.9 to 10.7 mAP and help 13 of 20 domains: like text prompts, they carry domain signal across model versions, although most of their utility stays tied to the source model.

4.5 A STRONGER BASELINE AND A SECOND BACKBONE

Two checks probe whether the recipe depends on Roboflow20-VL's low baselines or on Qwen3-VL. On LVIS Rare 50, whose long-tail classes have ordinary English names and whose frozen baseline

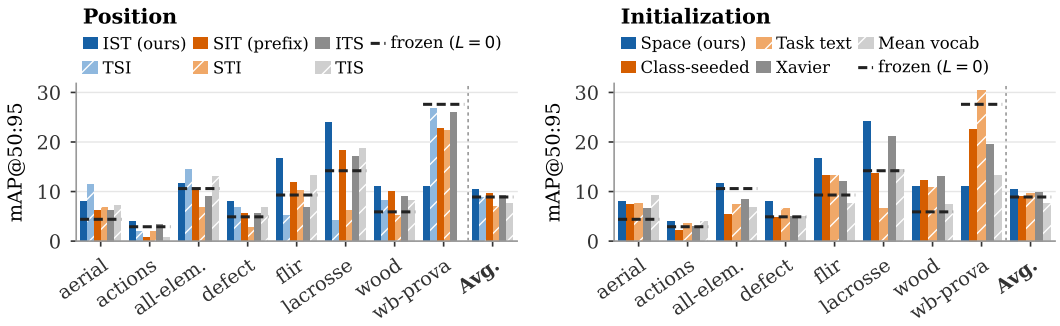

Figure 5: **Cross-modal placement and a space-token start give the most reliable soft prompts.** Test mAP@50:95 on the eight above-floor pool datasets and their nine-dataset average (Avg.); dashed lines mark the frozen $L{=}0$ model. *Left:* IST is best on 5 of 8 datasets and never the worst ordering; the NLP-style prefixes (orange) win none, and TSI collapses on lacrosse ($24.0 \rightarrow 4.1$). *Right:* the space-token start is best on 5 of 8, while semantic seeds collapse on individual datasets (task text on lacrosse, class names on all-elements). On wb-prova the frozen model (27.6) beats every learned configuration except task-text initialization. Values in Appendix A.

(22.4 mAP) is roughly twice Roboflow20-VL's, SOFTPROMPT with IST and $L{=}1$ reaches 25.7, a $+3.3$ gain comparable to its $+3.4$ on Roboflow20-VL; $L{=}3$ falls to 20.7, below the frozen model, so the $L{=}0$ control matters here too (Appendix B). Trained fresh on the unrelated Gemma-4-12B-it, with every configuration chosen on held-out support images only, the soft prompt beats the frozen baseline on 5 of 8 above-floor pool datasets ($+0.8$ mAP on average). Gemma prefers TSI on most of them, and its held-out margins do not reliably predict test gains (Appendix I).

4.6 BEYOND DETECTION: PROMPTING A ROBOT POLICY

We test whether placement also matters for a frozen $\pi_{0.5}$ vision-language-action policy on three RoboCasa tasks with ten demonstrations each (Appendix E). A prefix-only prompt reaches the action expert only indirectly and fails (1.7% success on PickPlace). A dual prompt that also enters the action-expert input raises the two near-floor tasks from 5.0% to 23.3% and 31.7%, comparable to LORA (21.7–26.7%) with roughly $400\times$ fewer trainable parameters. On OpenDrawer, which the base policy already performs, the soft prompt overrides the competent policy while LORA improves it, marking the recipe's boundary: a soft prompt must sit in the gradient path of the output it steers, and pays off where the frozen model is weak.

5 ABLATION STUDY

We isolate position, initialization, and length on the fixed nine-dataset pool. All win counts exclude dentalai, the floor control, and all values are mAP@50:95 from single runs.

5.1 INJECTION POSITION

IST is best on five of eight above-floor datasets, improves over the frozen baseline on seven, and is never the lowest-scoring ordering above the floor (Figure 5, left); it also has the best nine-dataset average (10.5, vs. 9.6 for the NLP prefix SIT and 6.9 for STI). Neither soft-first prefix wins an above-floor dataset. The strongest exception is informative: TSI wins aerial-airport, a single-class task, while on the multi-class sports dataset lacrosse it falls from IST's 24.0 to 4.1. Thus IST is the robust default, with TSI worth testing when the text query itself identifies the single target concept.

5.2 INITIALIZATION AND LENGTH

The space token wins five of eight above-floor datasets and yields the best average (10.5, vs. 9.9 for Xavier and 9.6 for task text; Figure 5, right). Semantic starts are less reliable: task-text initialization

falls to 6.6 on lacrosse versus 24.1 for space, while class-seeded initialization falls to 5.3 on all-elements versus 11.6. This supports a conservative start that preserves the existing query rather than injecting an unvalidated semantic prior.

No universal length works. Selected prompts use one to three tokens, but the sweep includes $L \in \{0, 1, 2, 3, 4, 8\}$ because a mis-sized prompt can underperform the frozen $L = 0$ control. Length and position also interact: aerial degrades with longer IST prompts but improves under TSI. The practical recipe is therefore a short validation sweep that always includes the unadapted model.

Finding. Where the tokens sit and what they start from decide whether soft prompting works: IST wins 5 of 8 above-floor datasets and is never worst, both NLP-style prefixes win none, and a space-token start avoids semantic seeds that collapse on individual datasets (task text 6.6 vs. 24.1 on lacrosse).

6 DISCUSSION

When does asking suffice? Ten images often suffice to steer knowledge a general VLM already encodes; the medical failure shows the converse, where neither a tiny prompt nor a low-rank adapter recovers absent knowledge. This boundary suggests a staged adaptation strategy: begin with a frozen-model prompt, which is cheap to store and sufficient when the failure is one of elicitation; escalate to a larger adapter when the prompt plateaus, and to a specialist when every generalist method stays at the floor. The 14.2 parity is between single-run aggregate scores, not evidence that the two methods are interchangeable on every domain; it shows that a tiny gradient-trained input belongs in the comparison as a serious first-stage baseline.

Practical recipe. Start with IST, space initialization, and $L \in \{0, 1, 2, 3, 4, 8\}$; select on held-out detection mAP rather than language-model loss. Treat TSI as a targeted alternative for single-class or text-dominant domains.

Limitations. The main limitation is that soft prompts are harder to train than LoRA: SoftPrompt's seed-to-seed spread averages 7.9× LoRA's at the accuracy-matching rank, while re-evaluating a fixed checkpoint barely moves its score (Appendices R and R.2). The spread therefore arises in training: with only ten images and a few thousand parameters, different seeds settle in different local optima, whereas LoRA trained on the same ten images is stable. Since the main comparison uses mostly single runs, differences on individual domains should be read with this variance in mind. On Gemma-4 the recipe still helps, but holdout-based selection is less reliable and TSI is preferred on most pool datasets, so the best boundary ordering is backbone-dependent (Appendix I). Finally, Soft2Hard creates an inspectable instruction, not a guaranteed faithful explanation of the continuous representation.

7 CONCLUSION

Soft prompting fills the gap between discrete prompt search and larger gradient-trained adapters. With cross-modal placement and meaning-preserving initialization, one to three learned tokens match the best LoRA rank on Roboflow20-VL at roughly 24,000× fewer trainable parameters, and they behave like prompts rather than weights: they leave the backbone unchanged, transfer across model versions, and can be verbalized into editable text. When the frozen VLM contains the relevant knowledge, few-shot adaptation becomes a matter of learning how to ask.

Ethics statement. The method reduces task-specific storage and supports human inspection through editable distilled prompts, but does not itself guarantee faithful explanations, regulatory compliance, or freedom from the risks of object detection systems, including surveillance misuse.

Reproducibility statement. Appendices G–H report settings and per-domain configurations; the remaining appendices record ablations, seed checks, controls, aggregate supports, best-of-position results, and qualitative results. Code is provided in the supplementary material, and code and prompt checkpoints will be released publicly.

AI use disclosure. We used generative AI tools to help write code and to proofread and edit the paper text. All AI-assisted code and text were reviewed and checked by the authors, who take full responsibility for the content of this work.

REFERENCES

Lakshya A Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziems, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J Ryan, Meng Jiang, Christopher Potts, Koushik Sen, Alexandros G Dimakis, Ion Stoica, Dan Klein, Matei Zaharia, and Omar Khattab. GEPA: Reflective Prompt Evolution Can Outperform Reinforcement Learning. In *International Conference on Learning Representations (ICLR)*, 2026.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. In *Robotics: Science and Systems (RSS)*, 2025.

Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale N Fung, and Steven Hoi. InstructBLIP: Towards general-purpose vision-language models with instruction tuning. *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Gautam Rajendrakumar Gare, Neehar Peri, Matvei Popov, Shruti Jain, John Galeotti, and Deva Ramanan. DetPO: In-Context Learning with Multi-Modal LLMs for Few-Shot Object Detection. In *European Conference on Computer Vision (ECCV)*, 3 2026. URL http://arxiv.org/abs/2603.23455.

Agrim Gupta, Piotr Dollár, and Ross Girshick. LVIS: A dataset for large vocabulary instance segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In *International Conference on Machine Learning (ICML)*, 2019.

Edward Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations (ICLR)*. International Conference on Learning Representations, ICLR, 6 2022. ISBN 2106.09685v2. URL https://arxiv.org/pdf/2106.09685.

Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *European Conference on Computer Vision (ECCV)*, 2022.

Muhammad Uzair Khattak, Hanoona Rasheed, Muhammad Maaz, Salman Khan, and Fahad Shahbaz Khan. Maple: Multi-modal prompt learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

Brian Lester, Rami Al-Rfou, and Noah Constant. The Power of Scale for Parameter-Efficient Prompt Tuning. *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2021.

Baiqi Li, Zhiqiu Lin, Wenxuan Peng, Jean De Dieu Nyandwi, Daniel Jiang, Zixian Ma, Simran Khanuja, Ranjay Krishna, Graham Neubig, and Deva Ramanan. NaturalBench: Evaluating Vision-Language Models on Natural Adversarial Samples. *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2024. URL https://linzhiqiu.github.io/papers/naturalbench/.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2021.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual Instruction Tuning. *Advances in Neural Information Processing Systems*, 36, 2023. ISSN 10495258.

Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, and Lei Zhang. Grounding DINO: Marrying DINO with Grounded Pre-Training for Open-Set Object Detection. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 15105 LNCS:38–55, 3 2024. ISSN 16113349. doi: 10.1007/978-3-031-72970-6{_}3. URL https://arxiv.org/pdf/2303.05499.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Lam Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2022.

Matthias Minderer, Alexey Gritsenko, Austin Stone, Maxim Neumann, Dirk Weissenborn, Alexey Dosovitskiy, Aravindh Mahendran, Anurag Arnab, Mostafa Dehghani, Zhuoran Shen, et al. Simple open-vocabulary object detection. In *European Conference on Computer Vision (ECCV)*, 2022.

Soroush Nasiriany, Abhiram Maddukuri, Lance Zhang, Adeet Parikh, Aaron Lo, Abhishek Joshi, Ajay Mandlekar, and Yuke Zhu. Robocasa: Large-scale simulation of everyday tasks for generalist robots. In *Robotics: Science and Systems (RSS)*, 2024.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. $\pi_{0.5}$: A vision-language-action model with open-world generalization. In *Conference on Robot Learning (CoRL)*, 2025.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning Transferable Visual Models From Natural Language Supervision. In *International Conference on Machine Learning (ICML)*, 2 2021. doi: 10.48550/arxiv.2103.00020. URL https://arxiv.org/abs/2103.00020v1.

Peter Robicheaux, Matvei Popov, Anish Madan, Isaac Robinson, Joseph Nelson, Deva Ramanan, and Neehar Peri. Roboflow100-VL: A Multi-Domain Object Detection Benchmark for Vision-Language Models. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 5 2025. URL https://arxiv.org/pdf/2505.20612.

Taylor Shin, Yasaman Razeghi, Robert L. Logan, Eric Wallace, and Sameer Singh. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. *EMNLP 2020 - 2020 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, pp. 4222–4235, 10 2020. doi: 10.18653/v1/2020.emnlp-main.346. URL https://arxiv.org/pdf/2010.15980.

Tu Vu, Brian Lester, Noah Constant, Rami Al-Rfou, and Daniel Cer. Spot: Better frozen model adaptation through soft prompt transfer. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2022.

Mert Yuksekgonul, Federico Bianchi, Joseph Boen, Sheng Liu, Zhi Huang, Carlos Guestrin, and James Zou. Textgrad: Automatic "differentiation" via text. *arXiv preprint arXiv:2406.07496*, 2024.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, 2024.

Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to Prompt for Vision-Language Models. *International Journal of Computer Vision (IJCV)*, 2022a. URL https://github.com/KaiyangZhou/CoOp.

Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Conditional prompt learning for vision-language models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022b.

Appendix

A Full Core Ablations

All ablations use the fixed nine-dataset pool defined in §4.1. Dentalai is retained as a floor control and excluded from win counts; the reported averages include all nine datasets. Position and initialization are separate ablation runs, so nominally identical IST/space settings can differ by 0.1 mAP. For wb-prova, all cells use the controlled ablation pool's `max_pixels`= 6M setting; the per-domain headline instead uses its separately tuned 3M setting.

A.1 Injection position

Table 2: **Full injection-position ablation.** Test mAP@50:95 for all six orderings of image (I), soft tokens (S), and text (T). **Bold**: best position per dataset. †The wb-prova TIS run collapsed to 0.0.

Position	Description	aerial	actions	all-elements	defect-detection	dentalai	flir	lacrosse	new-defects	wb-prova	Avg
Frozen baseline ($L = 0$)		4.4	2.9	10.6	4.9	0.0	9.3	14.2	5.9	27.6	8.9
IST	Cross-modal boundary	8.0	**4.0**	11.7	**8.0**	0.0	**16.6**	**24.0**	**11.0**	11.0	**10.5**
TSI	Other boundary	**11.4**	1.9	**14.4**	6.7	0.0	5.1	4.1	8.2	**26.7**	8.7
ITS	Soft-last suffix	6.3	3.3	9.1	5.6	0.0	6.8	17.1	9.1	25.9	9.2
TIS†	Soft-last, modalities swapped	7.2	0.8	13.1	6.7	0.0	13.2	18.7	8.1	0.0	7.5
SIT	Soft-first NLP prefix	6.3	0.8	10.4	5.6	0.0	11.8	18.3	10.0	22.8	9.6
STI	Soft-first, modalities swapped	6.7	2.0	6.7	2.8	**0.2**	10.2	6.2	5.2	22.3	6.9

A.2 Prompt length

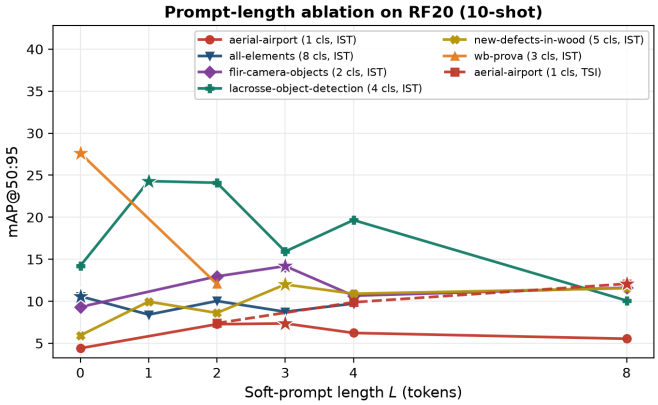

Figure 6: **Prompt length is small and dataset-dependent.** Available controlled test-mAP curves for $L \in \{0, 1, 2, 3, 4, 8\}$ under IST, plus the aerial-airport TSI curve. Stars mark each curve's best L; $L = 0$ is the frozen baseline.

A.3 Initialization

B Higher-Baseline Check: LVIS Rare 50

LVIS Rare 50 (Gare et al., 2026; Gupta et al., 2019) contains 50 long-tail classes whose names are ordinary English nouns. Its frozen baseline is 22.4 mAP, roughly twice the Roboflow20-VL baseline. SoftPrompt with IST and $L = 1$ reaches 25.7, a +3.3 gain. The result remains length-sensitive: $L = 3$ reaches only 20.7, below the frozen baseline, reinforcing the need for an $L = 0$ control.

Table 3: **Full initialization ablation.** A space token wins five of eight above-floor datasets and has the highest average. **Bold**: best initialization per dataset.

Initialization	Description	aerial	actions	all-elements	defect-detection	dentalai	flir	lacrosse	new-defects	wb-prova	Avg
Frozen baseline ($L = 0$)		4.4	2.9	10.6	4.9	0.0	9.3	14.2	5.9	27.6	8.9
Space	Single whitespace token	8.0	**3.9**	**11.6**	**8.0**	0.0	**16.6**	**24.1**	11.0	11.0	**10.5**
Mean	Mean vocabulary embedding	**9.2**	**3.9**	6.7	4.9	0.0	7.5	14.4	7.4	13.3	7.5
Xavier	Xavier uniform	6.7	3.0	8.4	5.0	0.0	12.1	21.2	**13.1**	19.5	9.9
Class-seeded	Class-name tokens	7.5	2.2	5.3	4.7	0.0	13.3	13.6	12.2	22.6	9.0
Task text	`detect` token	7.6	3.6	7.3	6.5	0.0	13.3	6.6	10.8	**30.3**	9.6

Table 4: **Soft prompting improves a stronger frozen baseline.** LVIS Rare 50, 10 shots per class, mAP@50:95.

Method	mAP@50:95	Δ
HARDPROMPT (frozen baseline)	22.4	–
SOFTPROMPT (IST, $L = 1$)	**25.7**	$+3.3$

C CROSS-TASK FORGETTING

This study measures how RF20 detection adaptation affects the model on other tasks. SOFTPROMPT never changes the backbone and, like any prompt, affects only the queries it is attached to, so other tasks are served by the unchanged model.

NaturalBench VQA. For each rank, we train 20 class-name LORA adapters, one independently on each RF20 domain, and evaluate each adapted model on NaturalBench (Li et al., 2024). We report the mean of the 20 per-adapter relative changes in strict group accuracy. At rank 64, averaging over the 20 adapters, strict group accuracy falls 35% relative ($37.1 \rightarrow 24.1$ absolute G-Acc); at rank 128 it falls 56% ($37.1 \rightarrow 16.3$). Forgetting therefore grows with rank.

RefCOCO grounding. The effect is not universal: a rank-4 LORA adapter trained on the-dreidel-project changes mean accuracy over eight RefCOCO splits by only -0.01 points (Table 13).

D PROMPT INTERPRETATION, DISTILLATION, AND TRANSFER

D.1 REPRESENTATION REGIMES WITH AND WITHOUT CLASS NAMES

We retrieve nearest neighbors of trained soft tokens in the text-vocabulary embedding matrix and in a cache of visual patch embeddings from the frozen encoder (Figure 3). When class names are withheld and the soft tokens must stand in for them, their nearest visual patches depict the target object. When class names remain in the query, matching the main protocol, the tokens do not align with object patches and instead appear to provide more general detection conditioning.

D.2 FULL SOFT2HARD RESULTS

Naively asking the model to explain an isolated soft token recovers nearby vocabulary artifacts rather than task meaning. SOFT2HARD instead preserves the context in which the prompt was trained: describe support images with the prompt active, summarize those descriptions, and evaluate the resulting instruction without soft tokens.

The 14.7 ceiling uses a test-selected best-of-IST/TSI prompt; the fixed-IST headline result is 14.2. The $+0.3$ mean is heterogeneous: the soft source helps on 10 of 20 datasets (mean $+1.9$) and hurts on 10 (mean -1.4), with the largest category gains on Aerial and Documents. The result supports editable verbalization as a useful interface, but not a claim that the text faithfully decodes each continuous dimension. Appendix Q isolates the same context dependence in a minimal arithmetic setting.

Table 5: **SOFT2HARD matches dedicated discrete prompt search, but most of the score comes from the describe-and-summarize recipe.** Category-level test mAP@50:95; Δ isolates the soft prompt's contribution by comparison with the no-soft-token control.

Method	Aerial	Doc.	F. & F.	Ind.	Med.	Sports	Other	All
SOFTPROMPT (test-selected best-of-position ceiling)	12.2	15.1	28.5	8.8	0.0	13.8	14.3	14.7
SOFT2HARD (soft source)	9.6	10.8	26.3	9.2	0.1	10.9	10.9	**12.5**
Base recipe (no soft tokens)	5.5	8.2	27.0	8.5	0.0	11.4	12.1	12.2
Δ (soft $-$ base)	$+4.1$	$+2.6$	-0.7	$+0.7$	$+0.1$	-0.5	-1.2	$+0.3$
GEPA (Agrawal et al., 2026)	6.3	10.8	22.4	8.5	0.1	12.8	11.5	11.6
DetPO (Gare et al., 2026)	8.6	10.1	26.7	9.9	0.1	11.0	10.5	**12.5**

D.3 TRANSFER ACROSS MODEL VERSIONS

Prompts trained on Qwen3-VL-8B can be loaded without projection or retraining into Qwen3.5-9B. The transferred prompt reaches 10.7 mAP against the target model's 9.9 baseline, helping 13 of 20 datasets and hurting seven; it remains 4.0 points below its 14.7 origin-model score. Thus some domain signal transfers, but much remains tied to the origin embedding geometry. Table 14 reports the category breakdown.

Table 6: **Naive cross-version transfer recovers part of the adaptation.** Qwen3-VL-8B to Qwen3.5-9B, all-domain mAP@50:95.

Method	All-domain mAP	Δ
Qwen3.5-9B baseline	9.9	–
9B + transferred prompt	10.7	$+0.8$
8B origin (test-selected best-of-position)	14.7	$+4.8$

E BEYOND DETECTION: PROMPTING A ROBOT POLICY

We test whether placement also matters for a frozen $\pi_{0.5}$ vision-language-action policy (Black et al., 2025; Physical Intelligence et al., 2025). Three RoboCasa tasks (Nasiriany et al., 2024) use ten demonstration episodes each and 20 held-out closed-loop trials per seed over three seeds. A prefix-only prompt conditions PaliGemma but reaches the action expert only indirectly. A dual prompt adds tokens directly to the action-expert input and therefore places trainable conditioning in the path producing the flow-matching output.

Table 7: **Placement also matters for a robot policy.** Success rate over three seeds; **bold** marks the best prompt-space and adapter result per task.

Adapter	Trainable	PickPlace	Faucet	OpenDrawer
$\pi_{0.5}$ base	–	5.0%	5.0%	30.0%
Soft prompt, prefix-only	16K	1.7%	–	–
Soft prompt, dual	49K	**23.3%**	**31.7%**	21.7%
Soft prompt, dual, early-stopped	49K	–	–	**45.0%**
LORA, PaliGemma-only	19.6M	23.3%	21.7%	66.7%
LORA, + action expert	26.5M	**26.7%**	**26.7%**	**78.3%**

The dual prompt raises the two near-floor tasks from 5.0% to 23.3% and 31.7%, matching or exceeding the coverage-matched LORA result. On OpenDrawer, where the base already reaches 30.0%, the full-strength prompt falls to 21.7% and early stopping reaches 45.0%, while LORA reaches 66.7–78.3%. This is a mechanism study over three tasks, not a broad robotics benchmark claim.

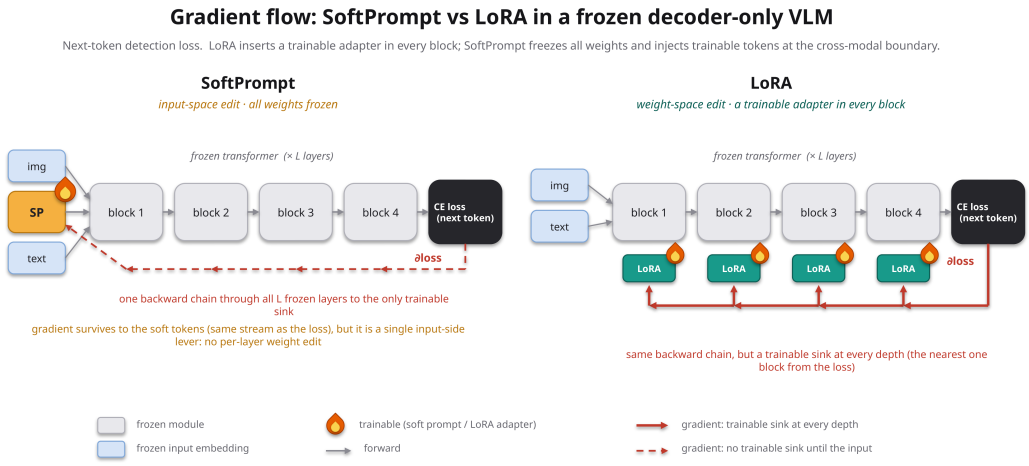

Figure 7: **A soft prompt is a weaker optimisation lever than LoRA: the same backward chain, but a single trainable sink at its far end.** Gradient flow for the two adaptation methods on a frozen decoder-only VLM trained with the next-token detection loss. Gray boxes are *frozen* modules and blue boxes frozen input embeddings; a flame marks a *trainable* parameter (the amber soft-prompt tokens on the left, the teal LoRA adapters on the right). Gray arrows are the forward pass; crimson arrows the backward chain from ∂loss, which traverses the same frozen blocks in both methods. **(Left, SoftPrompt)** the chain crosses all N_{layers} frozen layers before reaching the only trainable sink (dashed: no update until the input), the soft tokens injected at the cross-modal boundary. **(Right, LoRA)** the same chain feeds a trainable adapter at every depth (solid taps); the nearest sink is one block from the loss, and updates land at every layer. SoftPrompt trades this optimisation disadvantage for a much smaller task-specific parameter block; both methods keep the pretrained matrices frozen.

F Gradient Flow Comparison

Figure 7 contrasts the backward-pass optimisation signature of SoftPrompt against LoRA (§3.2): the same backward chain through every frozen block, but a single trainable sink at its far end rather than one at every depth.

G Implementation Details

SoftPrompt (detection). All experiments use PyTorch 2.x with `bfloat16` on NVIDIA A6000 GPUs. AdamW, lr 5×10^{-3}, cosine schedule, batch size 4, and space-token initialisation. Thirteen of the 20 headline runs use a 5-epoch budget; seven use a 50-epoch budget with early stopping (patience 15), as detailed in Appendix H. For the reported Qwen3-VL headline, L is selected per domain by test mAP from the short length sweep; this test-informed selection is disclosed in §4.1. The separate Gemma replication instead uses only a support-set holdout for all hyperparameter selection. Measured over all 226 ablation runs, the training loop's single-GPU wall-clock in the 5-epoch regime (excluding vLLM export and test evaluation) has median 2.0 hours (p10 7 minutes, p90 5.5 hours): the fastest domain (aerial-airport) trains in $\sim$4 minutes, while high-resolution domains (paper-parts) take up to 15 hours. Evaluation uses a vLLM engine with the trained soft-prompt rows appended to the embedding table as extended vocabulary.

LoRA SFT baseline (detection). Adapters on the language-model decoder's `q_proj`, `k_proj`, `v_proj`, `o_proj`, `gate_proj`, `up_proj`, and `down_proj` via `peft`; $\alpha = 2r$, no dropout; learning rate 5×10^{-5}, cosine schedule with 0.2 warmup ratio; effective batch size 2; 70 optimizer steps; bf16; seed 42. Appendix K reports the complete rank sweep and provenance.

Robotics ($\pi_{0.5}$ / RoboCasa). Base policy: $\pi_{0.5}$ pretrained with RoboCasa data (PandaOmron embodiment; three camera views, 16-dim state, 12-dim action). Tasks: `PickPlaceCounterToCabinet`, `TurnOnSinkFaucet`, `OpenDrawer`, each with its own K=10 per-seed demonstration subsets

and normalisation statistics. Soft prompt: learnable tokens on the PaliGemma prefix (prefix-only, $N{=}8$) or on both the prefix and the action-expert input (dual, $N{=}16$, $M{=}16$); IST format; the PaliGemma-side tokens use space-token initialisation as in §3.2, but the action-expert tokens are initialised randomly, since the action expert has no token vocabulary to seed from; trained with gradient accumulation to an effective batch of 128 and lr $2{\times}10^{-1}$ for the dual configuration, 2,000 steps. The early-stopped `OpenDrawer` variant halts the same dual configuration at 750 steps; the stopping point was selected on a seed-0 sweep over steps and learning rate, then re-run on 3 seeds. LoRA: openpi reference configuration, rank 16 on attention and FFN modules; the "+ action expert" variant restricts to the rank-16 adapters (26.5M), and the PaliGemma-only variant matches the prefix prompt's coverage (19.6M). Both arms train on the same $K{=}10$ demonstration episodes per seed. Evaluation: 20 closed-loop episodes per seed on freshly sampled held-out scene instantiations; the environment's random generator is re-seeded per episode index, so all arms face byte-identical scenes.

H Per-Dataset Configurations

Table 8 lists, for every Roboflow20-VL dataset, the frozen baseline, the selected SoftPrompt configuration behind Table 1's IST row (prompt length L, the epoch actually reached in training, and test mAP), and the best-of-position configuration (winning ordering, its L, and test mAP). The selected lengths are $L \in \{1, 2, 3\}$ (mean 1.75, *i.e.* 7,168 trainable parameters on average, range 4,096–12,288); TSI wins 6 of 20 datasets, concentrated on single-class and text-dominant domains (§5.1). All other hyperparameters are shared across datasets (Appendix G). The IST row uses dataset-specific budgets: 13 of 20 datasets train for 5 epochs (matching the §5 ablation setting), and 7 train for 50 epochs with early stopping (patience 15).

Table 8: **IST wins 14 of 20 datasets; TSI takes the other 6, concentrated on single-class and text-dominant domains.** Per-dataset SoftPrompt configurations on Roboflow20-VL (10-shot, mAP@50:95). IST: the fixed-position row of Table 1 (per-domain L). **Ep**: epoch actually reached by the IST checkpoint. Best of IST/TSI: the per-dataset best ordering, selected on the test split only for this appendix ceiling; the fixed-IST headline does not use this selection.

				IST		**Best of IST/TSI**		
Dataset	**Category**	**Baseline**	**Ep**	L	mAP	Format	L	mAP
actions	Sports	2.9	5	3	3.2	IST	3	3.2
aerial-airport	Aerial	4.4	41	1	9.1	TSI	8	12.1
all-elements	Documents	10.6	15	1	13.4	TSI	1	14.4
aquarium-combined	F. & F.	29.2	7	1	35.7	IST	1	35.7
defect-detection	Industrial	4.9	5	1	6.3	TSI	1	6.7
dentalai	Medical	0.0	3	2	0.1	IST	2	0.1
flir-camera-objects	Other	9.3	5	2	16.6	IST	2	16.6
gwhd2021	F. & F.	1.3	21	1	4.2	TSI	1	5.8
lacrosse-object-detection	Sports	14.2	5	1	24.3	IST	1	24.3
new-defects-in-wood	Other	5.9	5	3	12.0	IST	3	12.0
orionproducts	Other	3.1	5	3	6.0	IST	3	6.0
paper-parts	Documents	4.4	5	2	15.9	IST	2	15.9
recode-waste	Industrial	17.0	5	1	16.7	IST	1	16.7
soda-bottles	Other	15.3	23	1	14.2	IST	1	14.2
the-dreidel-project	Other	12.1	1	3	22.9	IST	3	22.9
trail-camera	F. & F.	41.9	5	2	41.9	TSI	2	43.5
water-meter	Industrial	2.6	5	1	2.9	IST	1	2.9
wb-prova	F. & F.	27.6	5	2	28.8	IST	2	28.8
wildfire-smoke	Aerial	8.3	41	1	8.9	TSI	1	12.3
x-ray-id	Medical	0.1	5	3	0.0	IST	3	0.0
Mean		10.8		1.75	14.2			14.7

I GEMMA-4 GENERALISATION: SEARCH METHODOLOGY AND FULL RESULTS

We train the method fresh on `Gemma-4-12B-it`, a VLM family unrelated to Qwen3-VL, and find that while the recipe generalises (beats baseline on 5 of 8 above-floor pool datasets), its support-holdout selections do not reliably predict test performance. This appendix gives the search methodology and the complete per-dataset accounting behind that finding.

Search methodology. For each dataset, using only its 10-shot support set (never test), we hold out a fixed fraction as an internal validation split and run support-holdout successive halving over five stages: (S1) a format $\times$ length screen ($\{\text{IST, TSI}\} \times L \in \{1, 2, 3\}$, plus an $L{=}0$ control, space initialisation, lr $5{\times}10^{-3}$, no gradient accumulation); (S2) a gradient-accumulation sweep (accumulation over 4 steps or the full support set) on the top-2 S1 configurations; (S3) a learning-rate refinement ($\{1,3,10\}{\times}10^{-3}$) of the best config so far; (S4) an initialisation check (`xavier`, `mean`) against space initialisation; and, added after the first pass on Qwen3-VL showed a wider net was needed for this backbone, (S5) an epoch-budget refinement (10, 15 epochs against the default 5) and an extended length screen up to $L{=}5$ for two datasets (defect-detection, flir-camera-objects) whose initial holdout margin over control was thin or, on a first attempt, entirely lost to transient GPU contention on a shared cluster node (confirmed by re-running in isolation). Every stage uses support-holdout mAP and never reads the test set. This is stricter than the test-informed length selection used for the exploratory Qwen3-VL headline (Appendix G). The selected configuration is then trained once more at full budget and evaluated on the true test split, as reported below.

Table 9: **The search's holdout margin does not predict its test margin.** Per-dataset search-selected configuration and full holdout-vs-test accounting. **Test (untuned)**: an $L{\in}\{1, 2, 3\}$ IST, space-init, lr $5{\times}10^{-3}$, no-accumulation config with no search at all, for reference. Accum: gradient accumulation steps (`f` = full support set per step). All mAP@50:95 test-split numbers are the best of two independent evaluations of the selected config (retrained from scratch; and the search-time checkpoint re-exported and re-evaluated); this test selection is used only for this appendix analysis. **Bold** marks the better of the selected and untuned test runs, not necessarily an improvement over the frozen baseline.

Dataset	L	Fmt	Init	LR	Accum	Hold. ctrl	Hold. sel.	Test base	Test sel.	Test (untuned)
actions	1	IST	space	5e-3	1	6.5	13.8	0.8	**1.0**	0.5
aerial-airport	1	IST	mean	1e-3	1	2.9	3.4	11.3	**11.4**	9.8
all-elements	3	TSI	space	5e-3	1	9.9	14.7	13.5	**17.4**	10.6
defect-detection	5	TSI	space	1e-3	4	2.8	13.7	6.6	3.7	**5.6**
dentalai (floor)	1	TSI	space	1e-3	f	0.9	9.9	0.5	0.6	0.4
flir-camera-objects	5	TSI	space	5e-3	4	11.7	16.5	14.0	**13.9**	13.7
lacrosse-object-detection	2	TSI	space	5e-3	f	9.0	18.5	11.3	10.5	**13.1**
new-defects-in-wood	2	IST	xavier	1e-3	f	8.9	13.2	3.7	3.9	**4.2**
wb-prova	3	TSI	space	1e-2	4	2.5	48.0	18.5	**24.3**	19.0

Two things follow from Table 9 that are not visible from the win/loss tally alone. First, on defect-detection, lacrosse-object-detection, and new-defects-in-wood, the search-selected configuration is not merely a weak win, it is beaten on test by the untuned default we never selected on anything: the search actively picks the wrong configuration for these three datasets, not just an insufficiently good one. Second, the size of the holdout margin carries no information about test reliability: wb-prova has the largest holdout margin in the pool (48.0 vs. 2.5 control, $19\times$) and the largest test gain ($+5.8$); defect-detection has the second-largest holdout margin (13.7 vs. 2.8, $4.9\times$) and the worst test result in the pool. A held-out mAP sweep over 10 support images per class is evidently enough signal to separate configurations that generalise to the model's own training distribution, but not always enough to predict which configuration the frozen backbone will prefer on unseen test images. We report this as an open reliability gap in the Gemma search procedure, not as evidence against the underlying method. The Qwen3-VL ablations are test-informed and therefore cannot establish an equivalent holdout-to-test selection claim.

J SERVING SOFT PROMPTS WITH STANDARD INFERENCE ENGINES

A practical objection to soft prompts is that they seem to require a custom forward pass (an embedding hook at the injection position), locking inference to a research codebase. Our evaluation pipeline shows this is not the case: a trained soft prompt can be served by an *unmodified* high-throughput engine, vLLM in our case, through the model's own vocabulary.

The export works as follows. After training, the L learnt vectors are appended as L new rows of the backbone's input-embedding matrix, and L new special tokens (`<SP_0>`, `<SP_1>`, ...) are registered in the tokenizer, one per row; the result is written as a standard checkpoint. The base weights and the original embedding rows are untouched: the model gains L rows and loses nothing. At inference, the detection query simply contains the placeholder string `<SP_0>`... at the chosen injection position (*e.g.* between the image and the text for IST), which tokenises to exactly the L new ids (verified at load time); the engine embeds them like any other tokens, which *is* the soft prompt. No engine modification, no custom hooks, and all of the engine's optimisations (paged attention, tensor parallelism, batched generation) apply unchanged.

Three consequences are useful in practice. First, one exported engine serves both arms of any soft-vs-base comparison: a query that omits the placeholder tokens is processed exactly as by the base model, which is how the distillation baseline of §4.3 shares a single engine with the soft-prompted arm. Second, the marginal cost of a domain is L embedding rows, so shipping an adapted domain is a vocabulary patch rather than a model copy: Table 10 measures this directly from the checkpoint files rather than from parameter counts alone, since on-disk size also reflects serialization overhead that a raw parameter count does not. Third, switching domains requires selecting only a small span of prompt embeddings rather than a much larger set of per-layer low-rank factors. Both are removable task-specific components; our reference implementation (`softprompt_pkg`, released with the paper) demonstrates switching soft prompts by detaching one prompt hook and attaching another without changing the pretrained backbone matrices. The same property extends to the exported checkpoint above: because each domain occupies a disjoint, fixed span of token ids, multiple domains' rows can coexist in one resident vocabulary, and a request activates whichever domain's tokens it includes. All main-paper detection evaluations use this path. Code, including the export and evaluation pipeline, will be released.

Table 10: **The trained checkpoint is proportionally as small as the parameter count suggests, measured, not estimated.** On-disk size of the trainable adapter only (no base-model weights): SOFTPROMPT's `torch.save` tensor vs. LORA's `adapter_model.safetensors`, both at their native training precision (fp32). SOFTPROMPT sizes are exact measurements at each L; the mean-L=1.75 row interpolates linearly (each additional token costs a fixed 16 KB, $= d\times 4$ bytes for d=4096), matching the per-dataset L mix used in Table 1. Ratio: LORA size / SOFTPROMPT mean-L size. **Bold**: r=64, the rank that matches SOFTPROMPT's accuracy (Table 1).

Checkpoint	Size	Ratio vs. SOFTPROMPT (mean L)
SOFTPROMPT (L=1)	17.6 KB	–
SOFTPROMPT (L=2)	33.6 KB	–
SOFTPROMPT (L=3)	49.6 KB	–
SOFTPROMPT (mean L=1.75)	29.6 KB	1×
LORA (r=4)	41.7 MB	1,442×
LORA (r=8)	83.3 MB	2,882×
LORA (r=16)	166.6 MB	5,762×
LORA (r=32)	333.1 MB	11,522×
LORA (r=64)	**666.1 MB**	23,041×
LORA (r=128)	1.3 GB	46,080×

The storage ratio at r=64 (23,041×) is close to but slightly below the raw parameter-count ratio ($174.6\text{M}/7{,}168 \approx 24{,}357\times$, §3): both checkpoints are fp32, so the gap is entirely serialization overhead, a fixed `torch.save` pickle header that is negligible against LORA's multi-megabyte files but not against SOFTPROMPT's few-KB ones. We report the measured figure rather than the parameter estimate because it is what a deployment actually pays.

K FULL LORA RANK SWEEP

Table 11 reports both LoRA variants at every rank. The + *instructions* rows use one multi-class query that appends each class's dataset description and annotator instructions to the class names (§4.2); SOFTPROMPT has no counterpart in that setting yet (an instruction-augmented soft prompt is future work). The main comparison (Table 1, Figure 1) therefore uses the default class-name protocol, while Table 1 includes the best instruction-augmented row only as an unmatched reference and the full sweep appears here. For reference, SOFTPROMPT with a per-dataset, test-selected choice of IST/TSI reaches 14.7; this appendix ceiling is distinct from the fixed-IST 14.2 headline.

LORA training configuration. Every rank is trained per-dataset with LLaMA-Factory (Zheng et al., 2024) on Qwen3-VL-8B, target modules `q_proj`, `k_proj`, `v_proj`, `o_proj`, `gate_proj`, `up_proj`, `down_proj` (attention and MLP), α=2r (e.g. α=128 at r=64), dropout 0, AdamW, learning rate 5e-5, cosine schedule with 0.2 warmup ratio, effective batch size 2 (per-device batch 1, gradient accumulation 2), 70 optimizer steps, bf16, and seed 42, matching this paper's own default training seed (§4.1). These settings were verified directly against the saved checkpoints' `adapter_config.json` and `training_args.bin`, rather than re-derived or estimated, and were identical across every sampled rank and dataset.

Table 11: **LORA rank sweep on Roboflow20-VL 10-shot (mAP@50:95): accuracy peaks at r=32–64 and declines at r=128.** Both variants (class names only; with per-class instructions). Figure 1 plots the class-name sweep used for the matched main comparison; Appendix C separately reports cross-task interference while each detection adapter remains active. **Bold**: best configuration per variant.

Method	Params	Aerial	Doc.	F. & F.	Ind.	Med.	Sports	Other	All
LoRA (r=4)	10.9M	8.1	8.1	27.0	8.8	0.0	11.9	10.5	12.1
LoRA (r=8)	21.8M	11.3	7.3	28.7	8.3	0.2	10.3	11.1	12.7
LoRA (r=16)	43.6M	12.3	13.8	28.1	6.6	0.3	8.6	10.9	12.8
LoRA (r=32)	87.3M	17.1	18.7	28.6	7.6	0.4	10.4	8.7	13.7
LoRA (r=64)	174.6M	20.2	19.9	28.6	7.1	0.7	9.6	9.4	**14.2**
LoRA (r=128)	349.2M	15.6	19.7	25.5	8.8	0.6	11.1	10.0	13.6
LoRA + instr. (r=4)	10.9M	8.5	9.4	26.9	10.8	0.3	12.2	12.1	13.1
LoRA + instr. (r=8)	21.8M	10.8	9.7	26.8	9.6	0.9	14.8	12.4	13.5
LoRA + instr. (r=16)	43.6M	12.7	13.0	29.4	11.3	0.3	10.3	10.6	13.9
LoRA + instr. (r=32)	87.3M	15.0	17.8	27.9	12.8	0.5	10.8	11.8	**14.8**
LoRA + instr. (r=64)	174.6M	15.6	20.8	26.7	9.2	1.0	10.3	12.1	14.5
LoRA + instr. (r=128)	349.2M	14.7	19.0	27.7	7.4	0.7	12.4	10.9	14.0

L SINGLE-CLASS EVALUATION BASELINES

DetPO's strongest numbers use a single-class protocol: the VLM is queried once *per class* per image, which multiplies inference cost by the class count and is not comparable to the single-call multi-class protocol used by every method in Table 1. For reference, DetPO reaches 15.3 all-domain mAP under single-class evaluation and 17.5 with VQA-score reranking. VQA-score reranking is orthogonal to the prompt and applies equally to SOFTPROMPT; combining them is left for future work.

Table 12: **DetPO's single-class protocol reaches 17.5 mAP with VQA-score reranking, above SOFTPROMPT's 14.2 multi-class, at class-count× the inference cost.** Single-class evaluation (one class per VLM call; reference only, not cost-comparable to Table 1).

Method	Params	Aerial	Doc.	F. & F.	Ind.	Med.	Sports	Other	All
DetPO (Gare et al., 2026)	0	8.3	19.1	30.3	13.7	0.1	14.2	12.2	15.3
DetPO + VQA Score	0	12.3	24.2	32.3	13.5	0.2	17.2	14.3	17.5

M CROSS-TASK GROUNDING: FULL REFCOCO RESULTS

Table 13: **A low-rank detection LoRA adapter causes negligible cross-task interference on grounding.** RefCOCO-family acc@0.50 (%) while the detection adapter remains active. Δ is LoRA (fine-tuned on the-dreidel-project, r=4) minus the frozen backbone. The SOFTPROMPT row omits its detection prompt and is therefore the base computation; deactivating the LoRA adapter likewise restores the base.

Method	RefCOCO val	RefCOCO testA	RefCOCO testB	RefCOCO+ val	RefCOCO+ testA	RefCOCO+ testB	RefCOCOg val	RefCOCOg test	Mean
Frozen backbone	91.3	93.1	87.6	85.9	90.1	80.7	88.2	88.0	88.1
LoRA (dreidel, r=4)	91.3	92.9	87.5	85.8	89.9	80.8	88.3	88.1	88.1
Δ (LoRA − base)	+0.00	−0.10	−0.10	−0.06	−0.16	+0.10	+0.10	+0.18	−0.01
SOFTPROMPT (prompt omitted)	≡ frozen backbone								

N CROSS-MODEL TRANSFER: PER-CATEGORY RESULTS

Table 14: **Transfer helps most on Sports (+5.3) and Documents (+3.3), and hurts on Industrial (−2.9).** Cross-model transfer of trained soft prompts per Roboflow20-VL super-category (Qwen3-VL-8B → Qwen3.5-9B, 10-shot, mAP@50:95). The 8B-trained prompt is loaded into Qwen3.5-9B with no retraining or projection.

Method	**Aerial**	**Doc.**	**F. & F.**	**Ind.**	**Med.**	**Sports**	**Other**	**All**
Qwen3.5-9B baseline	8.8	7.4	26.0	4.3	0.1	4.5	8.1	9.9
9B + transferred prompt	9.0	10.7	25.6	1.3	0.1	9.8	9.9	10.7
8B origin (upper bound)	12.2	15.1	28.5	8.8	0.0	13.8	14.3	14.7
Δ vs. 9B baseline	+0.2	+3.3	−0.4	−2.9	+0.1	+5.3	+1.8	+0.8
Δ vs. 8B origin	−3.2	−4.4	−2.8	−7.5	+0.1	−4.0	−4.5	−4.0

O DISTILLATION VS. DIRECT TRANSFER: PER-DATASET RESULTS

Table 15 compares two transfer forms for the same Qwen3-VL-8B-trained soft prompt: raw embeddings (direct, no retraining) and the §4.3 hard prompt distilled from it (best of the global/per-class variant, no soft tokens).

The two datasets where distillation wins (defect-detection, gwhd2021) are both near-floor swaps below 4 mAP either way; every dataset with a meaningful gap favours direct transfer, up to +9.9 on lacrosse-object-detection.

P TOKEN-SUBSTITUTION PROBE: DISCRETISING THE TRAINED PROMPT

If the trained soft tokens have meaningful discrete neighbours (§D.1), those neighbours should work as a prompt. This probe tests that directly: at evaluation time, each trained soft token is replaced by its nearest neighbour (cosine, top-1) in the model's own representation spaces, either the text-vocabulary embeddings (`text`) or the cached frozen-encoder patch embeddings (`patch`), and the discretised prompt is evaluated in place of the trained one. No retraining is involved; each dataset uses its canonical checkpoint.

Two observations (Table 16). First, text substitution stays above the frozen baseline on 6 of 8 above-floor pool datasets, all but actions and wb-prova†, which happen to be the two datasets where the trained prompt's own gain over baseline is smallest to begin with (+1.1 and +1.2 mAP, versus +3.1 to +9.8 elsewhere), leaving the least room for a discretised approximation to still clear the baseline. Where it does clear the baseline, text substitution retains roughly 33% of the trained prompt's gain in aggregate across those six datasets. The learned directions therefore point at genuinely useful discrete tokens, the functional counterpart of the retrieval reading in §D.1 and of the distillation result in §4.3, though the effect is a partial recovery rather than a free one. Second, text substitution

Table 15: **Direct embedding transfer beats its own distilled description on 18 of 20 datasets.** Qwen3-VL-8B → Qwen3.5-9B, test mAP@50:95, 10-shot. "Distilled" is the better of the §4.3 global/per-class hard prompt, evaluated with no soft tokens; "direct" loads the trained embeddings as-is (as in Appendix N). Δ is direct − distilled.

Dataset	**Distilled**	**Direct**	Δ
actions	1.6	2.5	+1.0
aerial-airport	7.8	10.7	+3.0
all-elements	9.8	11.2	+1.4
aquarium-combined	21.2	27.4	+6.2
defect-detection	3.6	3.2	−0.4
dentalai	0.1	0.2	+0.1
flir-camera-objects	14.7	17.6	+2.9
gwhd2021	1.8	0.9	−0.9
lacrosse-object-detection	7.1	17.1	+9.9
new-defects-in-wood	3.7	7.4	+3.8
orionproducts	3.5	4.6	+1.1
paper-parts	6.1	10.2	+4.1
recode-waste	7.5	9.8	+2.3
soda-bottles	3.6	9.0	+5.4
the-dreidel-project	15.9	17.0	+1.0
trail-camera	46.3	46.4	+0.1
water-meter	0.4	0.7	+0.4
wb-prova	26.2	27.8	+1.6
wildfire-smoke	5.5	7.6	+2.1
x-ray-id	0.0	0.1	+0.0
Mean	9.3	11.6	+2.3

Table 16: **The trained prompt's discrete shadow carries most of its signal, more reliably in text than in image-patch space.** Substituting every trained token with its nearest text token keeps the prompt above the frozen baseline on 6 of 8 above-floor datasets and wins on every one of them; nearest-patch substitution is consistently weaker and collapses generation entirely (zero parseable detections) on three datasets. **Bold**: better substitution mode per dataset. dentalai is the floor control: neither the trained prompt nor its discrete shadow moves it. †wb-prova's nearest-patch substitution collapses generation entirely.

Dataset	Config	Baseline	Trained soft	Nearest `text`	Nearest `patch`
aerial-airport	1T TSI	4.4	11.4	**6.9**	6.2
actions	3T IST	2.9	4.0	**2.6**	0.0
all-elements	1T TSI	10.6	14.4	**11.1**	11.0
defect-detection	1T IST	4.9	8.0	**5.8**	5.7
dentalai	1T IST	0.0	0.0	0.0	0.0
flir-camera-objects	2T IST	9.3	16.6	**11.3**	0.0
lacrosse-object-detection	2T IST	14.2	24.0	**17.6**	17.0
new-defects-in-wood	2T IST	5.9	11.0	**8.7**	8.1
wb-prova†	2T IST	27.6	28.8	**25.1**	0.0
Mean		8.9	13.1	**9.9**	5.3

consistently outperforms patch substitution, winning on every above-floor dataset (pool means 9.9 vs. 5.3); on three datasets (actions, flir-camera-objects, wb-prova†), patch substitution collapses generation entirely to 0.0 mAP, zero parseable detections, while text substitution on the same three datasets still produces valid, if below-baseline on two of them, output. The useful discrete signal lives more reliably in the text-vocabulary space than in the visual-patch space.

Q A MINIMAL SOFT2HARD ILLUSTRATION: SYNTHETIC ARITHMETIC

The distillation result in §4.3 and the substitution probe above both study soft-to-hard behaviour inside the full RF20 vision pipeline, where the soft tokens, the image, and a class-name query are all present together. To isolate the phenomenon from that complexity, we built a minimal, non-visual companion: freeze a small causal LM (`Qwen3-VL-2B-Instruct`, used purely as a text model here) and train $L{=}10$ continuous soft tokens, in the same cross-modal-boundary-style placement (tokens immediately before the payload, §3.1), to solve a synthetic arithmetic task – given a short list of integers with no instruction text, predict their sum or product – then ask the trained model, in plain English, what the tokens mean. Full code and a runnable notebook reproducing every number below are released with the paper (`softprompt_pkg/soft2hard/soft2hard_synthetic_arithmetic.py` / `.ipynb`).

Table 17: **The soft prompt learns the synthetic task reliably across all four conditions.** Exact-match accuracy, 100 held-out examples, greedy decoding, 20 epochs, $L{=}10$.

Task	Exact match
Addition, 2-digit operands (0–99)	**1.00**
Addition, 1-digit operands (0–9)	0.96
Multiplication, 2-digit operands (0–99)	0.95
Multiplication, 1-digit operands (0–9)	0.86

Asked about the soft tokens in isolation (`"Elaborate: '<SP>'."`, no numbers), the model produces incoherent output in every condition: fragments of Hebrew, Greek, or other scripts unrelated to arithmetic. A nearest-vocabulary-token check, the same probe as Appendix P applied here, explains why: across all 4 conditions (40 soft-token positions checked in total), the single nearest vocabulary neighbour by cosine similarity is, without exception, the same rare musical-notation glyph (cosine sim. ≈ 0.80–0.82), regardless of task. This is not operation-specific signal, just where a lightly-trained 10-token prompt happens to sit in embedding space; the isolated query's incoherence is consistent with the model echoing something near that generic neighbour rather than reasoning about the span at all.

Asking about the tokens *in the context they were trained in* – immediately followed by a real number list, the exact shape seen during training – changes this completely. `"What is the task prompt of: '<SP>: 53, 93'?"` answers *"What is the sum of the two numbers: 53 and 93?"* ***Answer:*** $53 + 93 = \mathbf{146}$ (correct, on the 2-digit addition prompt), and *"Multiply 6 by 0."* ... $6 \times 0 = 0$. ***Answer: 0*** (correct, 1-digit multiplication). `"Rewrite in your own words: ..."` reliably names the operation (*"Multiply 53 by 93."*) without always attempting to solve it; `"Describe the task prompt of: ..."` is good but the least stable of the in-context templates, occasionally hallucinating an unrelated frame entirely. Templates that invite ad-hoc computation rather than description (`"Elaborate"`, bare `"Rewrite"`) more often get the arithmetic itself wrong on 2-digit multiplication (53×93 answered 4949 or 4989, not 4929) even while still correctly implying the operation – the exact trained template reaches 0.86–1.00 exact match (Table 17), but placing the same tokens inside different surrounding phrasing does not inherit that reliability. One further asymmetry: self-description accuracy does not track task-solving accuracy – the 1-digit addition prompt solves the task at 0.96 exact match but is the one condition where `"What is the task prompt of"` most often misreads it as a sequence-completion task rather than addition.

Finding. In a minimal setting stripped of vision entirely, a trained soft prompt is not self-describing in isolation (an isolated query only recovers the nearest generic vocabulary token, not task content), but it *is* soft-to-hard distillable once the query preserves the context it was trained in, at which point natural language recovers a real, often exactly correct, description of the task the tokens encode. This mirrors, in miniature, the RF20 result that verbalisation requires the tokens to be queried inside a working prompt, not in a vacuum.

R MULTI-SEED VARIANCE CHECK

Every ablation-table number in the main paper is a single run; the main text therefore treats per-dataset differences below roughly one mAP point cautiously. Each pool dataset's canonical Space/IST

checkpoint (the same configuration reported in §5) is retrained from two additional seeds (123, 2024) alongside the original (seed 42), with no other change.

Table 18: **Seed-to-seed spread averages 28% of the per-dataset mean, matching the 20–28% estimate the main paper's ablation tables are built on.** Test mAP@50:95 across three training seeds per pool dataset, canonical Space/IST configuration. CV: coefficient of variation (std / mean). dentalai is the floor control and is excluded from this variance summary; unlike this summary, core-ablation averages include all nine datasets.

Dataset	seed 42	seed 123	seed 2024	Mean ± Std	CV
aerial-airport	7.98	6.45	8.10	7.51 ± 0.92	12%
actions	3.94	5.40	3.93	4.42 ± 0.84	19%
all-elements	11.59	6.81	14.97	11.12 ± 4.10	37%
defect-detection	7.97	5.52	5.45	6.31 ± 1.43	23%
flir-camera-objects	16.55	16.96	7.00	13.50 ± 5.64	42%
lacrosse-object-detection	24.10	11.02	13.24	16.12 ± 7.00	43%
new-defects-in-wood	10.97	12.37	9.42	10.92 ± 1.47	14%
dentalai (floor control)	0.00	0.00	0.01	excluded	
wb-prova[†]	28.80	13.74	19.64	20.73 ± 7.59	37%

[†]wb-prova uses the per-domain `max_pixels`= 3M setting in this seed study; the controlled core-ablation tables use 6M.

The mean coefficient of variation across the eight fully-collected, above-floor datasets is 28% (range 12–43%), confirming the 20–28% disclosed in §4.1: single-run ablation numbers carry real seed noise, and per-dataset deltas below ~ 1 mAP should be read as ties rather than genuine effects. The spread is not uniform: aerial-airport and new-defects-in-wood are comparatively stable (12–14%), while all-elements, flir-camera-objects, lacrosse-object-detection, and wb-prova swing by 37–43% of their own mean. This does not track test-set size: flir-camera-objects has the pool's largest test split (500 images) and one of the highest CVs, while aerial-airport has the smallest (34 images) and the lowest, so whatever drives a given dataset's seed sensitivity is not simply "fewer test images."

R.1 The spread is training noise, not evaluation noise

Table 18 retrains from three seeds, so its spread conflates two sources: different random initializations reaching different local optima during training, and stochasticity in the vLLM generation step at test time. To isolate them, we take the seed-42 checkpoint for seven above-floor pool datasets and re-run test-set evaluation twice more, varying only the generation seed (`--skip_training`, no retraining).

Table 19: **Eval-time variance is near zero; the seed-to-seed spread in Table 18 comes from training, not evaluation.** Same checkpoint, three evaluation seeds, test mAP@50:95. CV here is $\leq 5\%$ on every included dataset, an order of magnitude below the 12–43% training-seed CVs.

Dataset	eval-seed A	eval-seed B	eval-seed C	Std
aerial-airport	7.98	7.39	7.36	0.35
actions	3.94	3.94	3.96	0.01
all-elements	11.59	11.76	11.86	0.13
defect-detection	7.97	7.97	7.97	0.00
flir-camera-objects	16.55	16.13	16.56	0.24
lacrosse-object-detection	24.10	23.99	23.99	0.06
new-defects-in-wood	10.97	11.00	11.00	0.02
dentalai (floor control)	0.00	0.00	0.00	excluded

Finding. Re-evaluating a fixed checkpoint under three generation seeds moves test mAP by at most 0.35 points on any included dataset (mean std 0.12), against 12–43% training-seed CVs on the same datasets (Table 18). The seed-to-seed spread this paper discloses is a training-time phenomenon: different soft-prompt initializations land in different local optima, not measurement noise at test time.

R.2 IS THE SPREAD SPECIFIC TO SOFTPROMPT?

The 10-shot regime itself could simply be noisy, in which case any adapter would show comparable run-to-run spread. To test this, we retrain the accuracy-matching LoRA configuration (r=64, §4.2) from the same three seeds, same nine pool datasets, identical data and hyperparameters (Appendix K) apart from `seed`, and evaluate test mAP with the same protocol used to produce every LoRA number in this paper.

Table 20: **SOFTPROMPT's training-seed spread is not a general property of 10-shot Roboflow20-VL training: it averages 7.9× larger than LoRA's at the same accuracy-matching rank, on 6 of 8 above-floor pool datasets.** Test mAP@50:95, three training seeds (42, 123, 2024), r=64.

Dataset	Std (mAP pts) SOFTPROMPT	LoRA	Ratio	SOFTPROMPT mean	LoRA mean
aerial-airport	0.92	1.84	0.5	7.51	11.19
actions[‡]	0.84	0.40	2.1	4.42	8.08
all-elements	4.10	0.37	11.0	11.12	13.44
defect-detection	1.43	0.48	3.0	6.31	10.12
flir-camera-objects	5.64	0.15	36.9	13.50	4.99
lacrosse-object-detection	7.00	1.42	4.9	16.12	13.56
new-defects-in-wood	1.47	2.12	0.7	10.92	9.51
wb-prova	7.59	1.80	4.2	20.73	27.75
dentalai (floor control)	0.01	0.04	–	0.00	0.38

[‡]actions' seed-42 LoRA checkpoint (`qwen3vl-8b-rank64/actions/multi_class`) is corrupted at the source (a 13.7MB `adapter_model.safetensors`, versus ∼ 698MB for every other dataset and seed; `safetensors` fails to deserialize it), so only seeds 123 and 2024 are available for this row.

Finding. LoRA at the accuracy-matching rank is markedly more stable across training seeds than SOFTPROMPT: on 6 of 8 above-floor pool datasets its std is 2.1–36.9× smaller, averaging 7.9× across all 8 (excluding dentalai's floor effect on both), and its eval-only variance is likewise negligible (Table 19), ruling out measurement noise as the explanation. Two datasets (aerial-airport, new-defects-in-wood) are exceptions where LoRA's spread instead exceeds SOFTPROMPT's. The dominant pattern nonetheless says the seed-to-seed spread disclosed throughout this paper is largely an artifact of SOFTPROMPT's tiny parameter count, not a property of 10-shot Roboflow20-VL training in general: a few thousand trainable prompt parameters land in more different local optima across random seeds than LoRA's 174.6M do.

S QUALITATIVE DETECTIONS

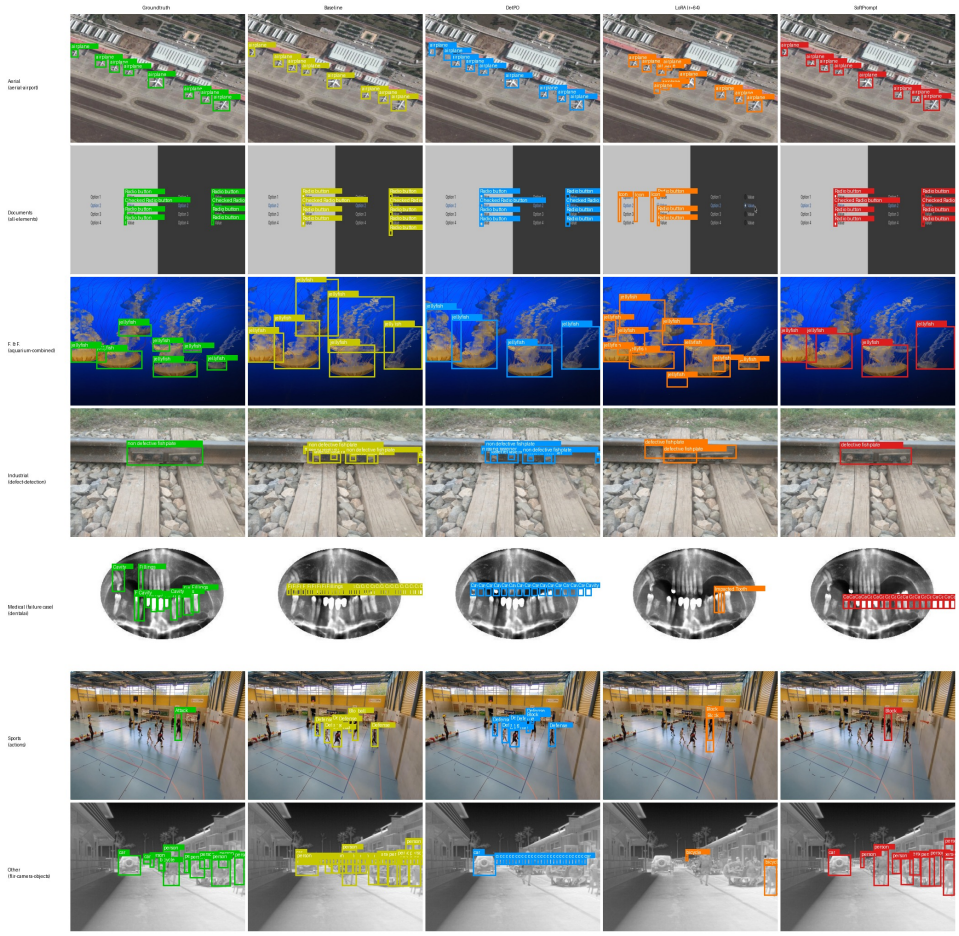

Figure 8: Qualitative detections on one held-out test image per super-category (7 of Roboflow20-VL's domains), comparing the frozen zero-shot baseline, DetPO, LoRA (rank 64), and SoftPrompt against groundtruth. The frozen baseline is the weakest column throughout, often over-generating or repeating a single label (dentalai, defect-detection); SoftPrompt localizes and labels cleanly on most domains but inherits the baseline's failure on dentalai, where no adapter recovers domain knowledge the frozen backbone never had.