RESCU: RELATIONAL SCENE CONSTRAINT UNDERSTANDING FOR ZERO-SHOT REFERRING IMAGE SEGMENTATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Referring image segmentation asks a model to find the exact pixels an expression describes. Zero-shot methods almost universally reduce this to cross-modal retrieval: embed the expression and each candidate region in a shared CLIP space and take the best match. That formulation cannot represent relational constraints—left_of, between, negation—because no single embedding-space comparison ever sees the joint configuration of target and anchor.

We introduce ReSCU (**Re**lational **S**cene **C**onstraint **U**nderstanding), which builds an explicit relational scene graph over open-vocabulary proposals, parses the expression into a typed query, and evaluates it symbolically to produce a *symbolic* hypothesis—then arbitrates it against a strong *appearance* hypothesis obtained by prompting a modern promptable-concept segmenter (SAM3) directly with the raw expression. The key idea, Dual-Hypothesis Mask Arbitration (DHMA), is to trust the appearance branch by default and promote the symbolic hypothesis only when a gate calibrated on held-out data says the parse is trustworthy. We study three such gates, of increasing capacity: hand-set thresholds, a sparse learned classifier, and a denser one over an expanded feature set.

The hand-set gate (**lock A**) sits at statistical parity with matched SAM3-text and does not transfer across datasets. A sparse logistic gate over 17 features (**lock C**), selected **blind** (grid fixed a priori, scored on held-out data, test touched once), produces a significant win on RefCOCOg (56.13/55.16 val/test mIoU, $\Delta{+}2.54$/$+1.89$pp, CI excluding zero) that transfers frozen to RefCOCO ($\Delta{+}0.91$/$+0.86$pp) and RefCOCO+ ($\Delta{+}1.17$/$+0.93$pp). Replacing lock C's classifier with a denser gradient-boosted one over 27 features (**lock D**) nearly doubles the margin (59.01/56.66 mIoU, $\Delta{+}5.41$/$+3.38$pp, CI $[+2.78, +4.02]$ test) and still transfers frozen to RefCOCO ($\Delta{+}1.53$/$+1.30$pp) and RefCOCO+ ($\Delta{+}2.19$/$+1.15$pp). Both gates lead every published zero-shot RIS number we could verify, including HybridGL (51.25/51.59) and RefChess (50.63/51.32), and both help all four expression types, not only the relational one lock A targeted.

An oracle upper bound sits 11.2 points above lock A's baseline; lock C closes $\approx 23\%$ of that gap and lock D $\approx 48\%$, so roughly half remains unrealized under the best gate we found. Detection is not the bottleneck—96.5% of gold targets survive to the proposal stage—but a frozen VLM used for spatial-edge prediction collapses to a single answer on $60/60$ held-out pairs, so our default edge backend stays geometric rather than learned. Lock D's larger gain also comes from intervening 3–$4\times$ more often through a harder-to-audit model, so we keep lock C as a sparse, interpretable alternative alongside lock D's headline result, and view closing the remaining oracle gap as the natural next step.

1 INTRODUCTION

Referring image segmentation (RIS)—producing a pixel-accurate mask for the region uniquely described by a natural language expression—requires both visual perception and relational language understanding. A human given "the mug to the left of the laptop" does not scan the image for mugs and guess: they build a mental map of the scene, locate the laptop, and identify the mug by its

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.

spatial relationship to that anchor. The relation, not the object category, is what disambiguates the target. Current zero-shot systems skip this step entirely, reducing relational reasoning to similarity matching—and that reduction, we argue, is the source of their fundamental limitations.

The core limitation. The dominant zero-shot framework Yu et al. (2023); Liu & Li (2025); Wang et al. (2025) generates candidate instance masks and selects the one with the highest CLIP cosine similarity to the expression:

$$M^* = \arg\max_{M_k} \cos\big(\phi(I, M_k),\ \psi(E)\big), \tag{1}$$

where ϕ denotes the CLIP (Radford et al., 2021) visual encoder applied to the masked/cropped region of image I defined by candidate mask M_k, ψ denotes the CLIP text encoder applied to expression E, and $\cos(\cdot, \cdot)$ is cosine similarity in the shared embedding space. For appearance-based expressions this holds up reasonably well. For the many RefCOCOg expressions that identify instances *by their relationships to other scene elements*, it breaks down for a structural reason, not a capacity one: the CLIP text encoder compresses the whole expression into a single vector, and the visual encoder scores a masked crop in isolation, without ever seeing the rest of the scene. Neither side of that comparison can natively evaluate "left_of", "between", "closest to," or "not the red one"—there is simply no place in the computation where the joint configuration of target and anchor is represented. HybridGL Liu & Li (2025) and IteRPrimE Wang et al. (2025) independently observe the symptom: CLIP-based methods struggle on directional or positional language. Modern promptable-concept segmenters such as SAM3 Carion et al. (2026) narrow the gap with a much stronger vision-language backbone—but, as we show in Section 4.2, do not close it, because the same structural gap persists: a purely appearance-driven system still cannot resolve which of several matching instances satisfies a stated spatial relation.

Our approach. ReSCU closes this gap on a simple premise: give the relational structure of the scene an explicit representation, evaluate it alongside—not instead of—a strong appearance hypothesis, and let a calibrated arbiter decide which one to trust on each image. Concretely, ReSCU constructs a relational scene graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ from open-vocabulary instance detections, parses the expression into a typed first-order query, and executes it symbolically to produce a *symbolic* candidate mask, while the raw expression is passed directly to SAM3's promptable-concept interface to produce an *appearance* candidate mask. Dual-Hypothesis Mask Arbitration (DHMA) then decides, per image, whether to trust the symbolic hypothesis or fall back to the generally stronger but relation-blind appearance hypothesis, using gates calibrated once on held-out data rather than a learned selector. We report this arbitration honestly: it is conservative by design, and our diagnostics (Section 4.4) show it currently realizes only a fraction of the available oracle headroom.

Contributions.

- We identify a structural gap that persists even for strong appearance-only segmenters: relational disambiguation among multiple concept matches requires information (explicit spatial relations between candidates) that a single text-to-mask or embedding-similarity call does not use.
- We introduce ReSCU's relational scene-graph pipeline: scene graph construction, a relational expression parser (REP), a symbolic query executor (SQE) with soft constraint scoring, and an optional relational prompt adapter (RPA).
- We introduce Dual-Hypothesis Mask Arbitration (DHMA), a calibrated, appearance-default arbitration mechanism between a symbolic and an appearance segmentation hypothesis, and compare three promote gates of increasing capacity: hand-set thresholds (**lock A**), a sparse learned classifier (**lock C**), and a denser gradient-boosted classifier over an expanded feature set (**lock D**).
- We show the promote decision, not the scene-graph pipeline, was the bottleneck: lock C and lock D, selected under an identical **blind** nested val-only protocol, are significant wins over matched SAM3-text on RefCOCOg and, applied completely frozen, transfer as significant wins to RefCOCO/RefCOCO+—where lock A does not transfer—while giving up interpretability and the type short-circuit; lock D roughly doubles lock C's margin by intervening 3–4$\times$ more often with a more complex model, so we report both.

- We provide a full diagnostic account of the pipeline (proposal/anchor recall, stage-wise failure attribution, an unreliable frozen VLM edge labeler) and an oracle headroom analysis showing lock D closes roughly half the gap, lock C a quarter, lock A about two percent.
- We report all of this with paired bootstrap significance testing and a leakage-safe, test-never-inspected selection protocol for all three locks (including a discarded, disclosed denser-gate attempt that improperly used transfer-dataset validation data, caught and rerun blind).

2 Related Work

Zero-shot RIS via CLIP similarity. GL-CLIP Yu et al. (2023) established the annotation-free RIS framework: generate candidate masks with a frozen proposal network, rank by CLIP global-local similarity. HybridGL Liu & Li (2025) (CVPR 2025) advances this with hybrid global-local feature extraction and a spatial guidance strategy for more robust region-text alignment. IteRPrimE Wang et al. (2025) (AAAI 2025) introduces iterative Grad-CAM refinement and a primary word emphasis module—yet its own analysis confirms CLIP-based zero-shot methods "always malfunction when encountering text inputs with positional information." Ref-Diff Ni et al. (2026) instead mines cross-modal structure from a frozen text-to-image diffusion model; CaR Sun et al. (2024) (CVPR 2024) recurrently filters an open vocabulary against a frozen CLIP backbone; CLIPSurgery Li et al. (2023) re-derives CLIP's internal attention for less noisy activation maps; and TAS Suo et al. (2023) (EMNLP Findings 2023), closest to our motivation, pairs a text-augmented matching score with a learned spatial rectifier. All five still work *within* the embedding-similarity paradigm of Eq. (1), improving the similarity signal rather than replacing it with an explicit, evaluable relational representation. ReSCU differs from all of the above in kind, not degree: rather than refining the similarity score, the symbolic branch replaces embedding retrieval with explicit predicate evaluation over a scene graph, and DHMA is the mechanism that decides, per image, whether that replacement is trustworthy.

Promptable-concept and foundation segmenters. SAM3 Carion et al. (2026) extends the Segment Anything line of work to *promptable concept segmentation*: given a short noun phrase, image exemplar, or combination, it detects, segments, and (in video) tracks every matching instance, roughly doubling prior open-vocabulary segmentation quality on Meta's own SA-Co benchmark. This makes SAM3 an unusually strong *appearance*-only baseline for RIS when its output is reduced to a single instance (Section 4.2), and we use it both as that baseline and as the mask decoder for both of ReSCU's hypotheses. It is, by construction, unable to resolve which of several instances matching a concept satisfies an additional stated spatial relation to another instance—the gap this paper targets.

Neuro-symbolic and program-based grounding. NS-VQA Yi et al. (2018) and the Neuro-Symbolic Concept Learner Mao et al. (2019) demonstrated that executing programs over symbolic scene representations outperforms end-to-end models for compositional questions. VisProg Gupta & Kembhavi (2023) showed that frozen LLMs can generate programs over visual APIs for zero-shot visual reasoning. NAVER (Cai et al., 2025) (ICCV 2025) is closest in spirit: a deterministic automaton where a frozen LLM emits ProbLog expressions executed probabilistically over detected entities, with self-correction retries and a no-entity fallback *within* that single automaton. ReSCU differs in kind: an explicit typed scene graph rather than automaton states, deterministic soft-constraint satisfaction (Eq. (3)) rather than probabilistic logic, and—most importantly—a fallback that is a competition *between two independently-computed hypotheses* rather than one pipeline retrying on itself.

MLLM-based segmentation. Supervised RIS moved from encoder-decoder architectures (LAVT (Yang et al., 2022), CRIS (Wang et al., 2022)) to large multimodal language models fine-tuned the same way: LISA Lai et al. (2024), GSVA Xia et al. (2024), PSALM Zhang et al. (2024a), SAM4MLLM Chen et al. (2024), OMG-LLaVA Zhang et al. (2024b), AnyRef He et al. (2024), and Text4Seg Lan et al. (2025) fine-tune large (multimodal) language models on (image, expression, mask) triplets, achieving strong supervised performance but requiring task-specific grounding labels that ReSCU avoids, and entangling language understanding with spatial grounding inside one model's attention, making it hard to attribute a failure to language versus grounding error—the attributability concern that partly motivates ReSCU's two-branch design: a wrong prediction traces to "symbolic promoted and wrong," "appearance kept and wrong," or "both agreed and wrong," unlike a single fine-tuned MLLM.

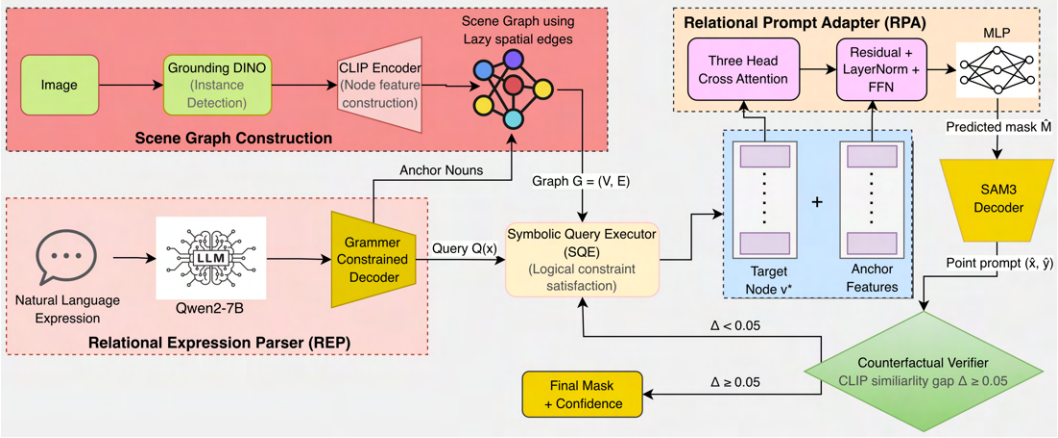

Figure 1: ReSCU pipeline. Scene-graph construction and relational parsing run in parallel and feed the symbolic query executor, which produces the *symbolic* hypothesis; the raw expression is also passed directly to SAM3's promptable-concept interface to produce the *appearance* hypothesis. Dual-Hypothesis Mask Arbitration (DHMA, Section 3.6) then decides, per image, whether to promote the symbolic hypothesis or keep the appearance default.

3 METHOD

3.1 PROBLEM REFORMULATION

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ denote a relational scene graph, where each node $v_i \in \mathcal{V}$ is a detected instance carrying an appearance embedding and a geometry embedding (Section 3.2), and each edge in $\mathcal{E}$ carries a typed spatial predicate $r \in \mathcal{R}$ drawn from a closed 12-predicate vocabulary. A referring expression E is parsed (Section 3.3) into a first-order logical query

$$Q(x) = T(x) \wedge \bigwedge_{k=1}^{K} R_k(x, a_k) \wedge \bigwedge_{j=1}^{J} \neg N_j(x), \tag{2}$$

where $T(x)$ asserts that x belongs to the target category, $R_k(x, a_k)$ constrains x relative to anchor node a_k, and $N_j(x)$ is a negation predicate. As a concrete instance, "the mug that is not red, to the left of the laptop" parses to $Q(\text{mug}) = T(\text{mug}) \wedge \texttt{left_of}(\text{mug}, \text{laptop}) \wedge \neg\, \texttt{red}(\text{mug})$. Symbolic grounding selects $v^*_{\text{sym}} = \arg\max_{v \in \mathcal{V}} \; s(v \mid Q, \mathcal{G})$ (Eq. (3), Section 3.4), which is one of the two hypotheses arbitrated by DHMA (Section 3.6); the other, M_{app}, comes directly from prompting SAM3 with the raw expression text. Figure 1 shows the complete pipeline.

3.2 SCENE GRAPH CONSTRUCTION

Instance detection and node features. Grounding DINO Liu et al. (2023) queried with `"objects . items . things ."` produces N instance proposals (threshold 0.30). Each node carries an appearance embedding $\mathbf{f}_i^{\text{app}} = \text{CLIP}_{\text{vis}}(\text{crop}(I, b_i)) \in \mathbb{R}^{768}$ (CLIP ViT-L/14, 10% margin crop) and a geometry embedding $\mathbf{f}_i^{\text{geo}} = \text{SinEnc}(b_i) \in \mathbb{R}^{128}$. On RefCOCOg test, this proposal set contains a box at IoU ≥ 0.5 with the gold target region for 96.5% of expressions (Table 9); detection is not the pipeline's binding constraint.

Relational edge construction: geometry by default. Edges between target candidates and the anchor set $\mathcal{A}$ identified by the REP can be labeled either by closed-form geometric rules over box coordinates (relative centroid position, containment, distance) or by prompting a frozen vision-language model with the candidate and anchor crops. Geometry is the default and the backend used for all reported numbers. We tested the VLM-labeling alternative directly and found it unreliable: on 60 held-out candidate–anchor pairs, three paraphrased prompts agreed with each other only as an artifact of the model collapsing to a single predicate—it predicted `left_of` on $60/60$ pairs regardless of true geometry, agreed with ground-truth box geometry on only 48.3% of pairs, and, most tellingly, produced the *same* predicate on $0/60$ pairs after we swapped candidate and anchor

crop order (which should flip most directional predicates)—i.e., its apparent stability is a collapsed prior, not genuine spatial reasoning (Table 8, Section 4.4). We report this as a negative result that directly motivates the geometric default, rather than omitting it.

3.3 Relational Expression Parser (REP)

The REP converts expression E into the formal query Q (Eq. (2)). Two backends are supported and both are reported here rather than only the more capable one: a **heuristic slot-filler** (pattern/dependency-based extraction of target noun, anchor(s), predicate(s), negation, and query type $S/R/N/C$), which is the backend used for the main reported numbers because it requires no LLM inference and is fully deterministic; and an optional frozen-LLM backend (Qwen2-7B Yang et al. (2024) with grammar-constrained JSON decoding via the `outlines` library) for expressions the heuristic backend cannot parse with confidence. No component is fine-tuned.

Expression taxonomy. **Type S** (simple): appearance-only ("the red mug"). **Type R** (relational): spatial predicates with anchors ("mug left of laptop"). **Type N** (negation): explicit negation ("chair that is not red"). **Type C** (compositional): chained relations or superlatives ("mug between laptop and keyboard"). Types are assigned by an *exclusive* partition with priority $N \succ C \succ R \succ S$ (an expression containing both a relation and a negation is scored as Type N, not double-counted), which is what makes the aggregate "All" row in Table 4 the exact size-weighted mean of the four type rows (Section 4.3); an earlier, non-exclusive version of this partition was the source of an arithmetic inconsistency in our original submission, which this revision corrects.

3.4 Symbolic Query Executor (SQE)

The SQE evaluates Q over $\mathcal{G}$ without any neural inference. It first filters candidates by type compatibility ($\cos(\mathbf{f}_i^{\text{app}}, \text{CLIP}_{\text{txt}}(T)) \geq 0.25$), then scores each by joint *soft* constraint satisfaction:

$$s(x \mid Q, \mathcal{G}) = s^{\text{cat}}(x) \cdot \prod_{k=1}^{K} s_k^{\text{rel}}(x)^{(1-\lambda_k^{\text{soft}})}, \tag{3}$$

where $\lambda_k^{\text{soft}} \in \{0, 1\}$ is the softness flag from the REP. Hard constraints can suppress candidates to zero; soft constraints add graded evidence rather than a hard pass/fail, which lets the executor rank plausible candidates even when an edge label is low-confidence. Superlatives are resolved as ordering operations over the candidate set; negation predicates remove violating candidates before final scoring. Replacing this soft scoring with the harder, threshold-based scoring used in an earlier version of the executor lowers the symbolic branch's own mIoU by roughly 6–7 points on a held-out 800-image slice (from $\approx 35.4/33.9$ to $41.9/39.8$ on val800/test800; Section 4.4), which is why soft scoring is used throughout. The top-scoring node's box is passed to SAM3 as a box prompt to obtain the symbolic branch's candidate mask $\hat{M}_{\text{sym}}$.

3.5 Relational Prompt Adapter (RPA):

An earlier version of this system used a lightweight cross-attention adapter (RPA, 6.2M parameters) to convert the selected node and its relational context into a SAM point prompt, trained on CC3M (image, caption) pairs with no segmentation annotation. We retain the RPA as an optional component—useful when a box prompt alone is not precise enough for a downstream point-based decoder—but all main results use SAM3's box-prompt interface directly on the SQE's selected node, with the RPA disabled. When enabled, its training target is the bounding rectangle of a *Grounding DINO detection box* on a CC3M image—not a human or ground-truth mask, at any training stage.

3.6 Dual-Hypothesis Mask Arbitration (DHMA)

Given the symbolic hypothesis $\hat{M}_{\text{sym}}$ from Section 3.4 and an appearance hypothesis $\hat{M}_{\text{app}}$ obtained by prompting SAM3 directly with the full expression text, DHMA decides which one to output: the appearance branch, backed by a modern foundation segmenter, is a strong default, and the symbolic branch should only override it when there is calibrated evidence the structural parse is trustworthy for this image—not simply because a symbolic query was successfully constructed. Concretely: (1) the SQE's top-k ranked candidate boxes are each decoded by SAM3 into a candidate mask; (2)

SAM3's text-prompt output on the full expression gives the appearance candidate; (3) a listwise scorer (following HybridGL's Liu & Li (2025) global-local scoring idea) ranks all candidates from both branches together; (4) the default output is the top appearance candidate; (5) the top symbolic candidate is **promoted** in its place only if its query type is Type R (relational expressions only), its self-consistency (IoU between the symbolic mask under two independently sampled scoring passes) is at least 0.22, it disagrees with the appearance hypothesis by an IoU margin of at least 0.40, and its promotion score (which folds in that disagreement) exceeds $\texttt{min_promote_score} = 1.55$. These gates were chosen via a nested, leakage-safe selection on RefCOCOg val alone (70%/30% fit/confirm, test untouched, Appendix J); the reselected configuration matches lock A, while broader type allow-lists that score higher on selection regress on confirmation and test—overfitting, not generalization. A post-hoc CLIP-similarity reject step (Section 3.8) is available but **disabled** here, since it discarded correct promotions at least as often as it caught incorrect ones. Under this policy the symbolic branch is promoted for roughly 2.8–3.0% of RefCOCOg images—DHMA is deliberately conservative, and Section 4.4 quantifies the cost.

3.7 Learned Promote Gates (Locks C and D)

Lock A's hand-set thresholds use three scalar features and restrict promotion to Type R by construction—interpretable, but, as Section 4.2 shows, it does not transfer. To test whether the ceiling is the *policy* or the *features*, we fit a logistic-regression gate over 17 features already computed by the two branches (raw and calibrated scores, score-gap variants, self-agreement, area ratio, per-branch deltas, one-hot expression type, expression length), allowing promotion for all four types (SRNC). Hyperparameters (ℓ_2, threshold τ, type allow-list, an optional "safe" gate) were chosen with a **blind nested protocol**: RefCOCOg val split 60/40 (fit/select) by a fixed seed, all 56 feasible grid points scored using *only* the 40% selection subset (no test field anywhere in the log), the winner (ℓ_2=0.1, τ=0.5, types SRNC, no safe gate) refit on full val, test touched exactly once (Appendix G documents the audit trail, including an earlier test-contaminated pass we discarded and reran blind). We call this **lock C**: it promotes $\approx 8\%$ of images, a continuous score rather than named thresholds—an interpretability loss weighed in Section 5.

Lock D: a denser gate over an expanded feature set. To test whether a more expressive model over richer features closes more of the oracle gap, we expand lock C's 17 features with three pairwise interaction terms and seven spatial-keyword indicators from the raw expression text, and replace the logistic model with a `HistGradientBoostingClassifier`, selected under the *identical* blind protocol as lock C (Appendix G) over a grid of six model families; the winner (depth-5 boosting, τ=0.475, types SRNC) promotes $\approx$ 25–28% of images, three to four times lock C's rate—**lock D**. An earlier attempt selected hyperparameters using a score that included RefCOCO/RefCOCO+ *validation* deltas alongside RefCOCOg's—not test leakage, but not a fair frozen-transfer comparison to lock C. We discarded that criterion and reran lock D blind to RefCOCO/RefCOCO+ entirely; the same configuration won again under the corrected objective, a verified coincidence, not a reused result (audit trail in Appendix G).

3.8 Counterfactual Verifier (secondary, reject-only signal)

Independently of DHMA's promotion gate, ReSCU retains a counterfactual verification mechanism that can flag a chosen mask $\hat{M}$ as suspect: a frozen LLM generates a counterfactual expression $\bar{E}$ describing the most visually similar wrong alternative, and the similarity gap

$$\Delta = \cos\big(\text{CLIP}_{\text{vis}}(\phi(\hat{M})),\ \text{CLIP}_{\text{txt}}(E)\big) - \cos\big(\text{CLIP}_{\text{vis}}(\phi(\hat{M})),\ \text{CLIP}_{\text{txt}}(\bar{E})\big) \tag{4}$$

is compared against a threshold θ. This is a bounded, pairwise discriminative check—closer to CLIP's native use case than the open relational ranking grounding itself requires (Eq. (1))—which is why CLIP can serve as a reasonable secondary check despite being a poor primary grounding signal. This reject step is off for the main reported system (Section 3.6) but kept available for settings where a conservative false-negative rate on promotions is acceptable.

Interpretability. For every prediction, ReSCU records which branch was output, which gates the symbolic branch passed or failed, and its constraint-satisfaction trace regardless of promotion—enabling the failure diagnosis in Section 4.5, unavailable from a single-branch appearance-only system.

Table 1: **Main result, RefCOCOg (UMD), matched mIoU (%) vs. SAM3-text.** Faster R-CNN detector fallback (Section 4; remount reproduces this within noise, Appendix F). Lock A is at statistical parity (CI includes 0). Lock C and lock D replace the thresholds with learned gates under an **identical blind** val-only protocol (Section 3.7)—both **significant** wins.

Method	val	test
SAM3-text (matched baseline)	53.60	53.27
ReSCU (DHMA, lock A)	53.81	53.29
ReSCU (DHMA, lock C, sparse)	56.13	55.16
ReSCU (DHMA, lock D, dense)	**59.01**	**56.66**
Δ lock A vs. SAM3-text	$+0.21$	$+0.02$
95% CI (paired bootstrap, n=10,000): includes 0 on both splits		
Δ lock C vs. SAM3-text	$+2.54$	$+1.89$
95% CI: $[+1.86, +3.23]$ (val), $[+1.42, +2.35]$ (test) — excludes 0		
Δ **lock D** vs. SAM3-text	$\mathbf{+5.41}$	$\mathbf{+3.38}$
95% CI (test): $[+2.78, +4.02]$ — **excludes 0**; val CI not computed (selection split)		

4 EXPERIMENTS

4.1 SETUP

Datasets. **RefCOCO** Kazemzadeh et al. (2014) (142k expressions, short, $\sim$50% positional) and **RefCOCO+** Yu et al. (2016) (141k, location words forbidden), both val/testA/testB; **RefCOCOg** Mao et al. (2016) (104k, avg. 8.4 words, highly relational, UMD partition, val/test)—our primary benchmark.

Baselines. *Matched (Table 1):* **SAM3-text**, feeding the expression's head noun phrase directly to SAM3's promptable-concept interface with no ReSCU pipeline, on identical splits and evaluation code—the comparison we can fully control. *Published zero-shot (Table 2, cited not re-run):* GL-CLIP Yu et al. (2023) (CVPR'23), IteRPrimE Wang et al. (2025) (AAAI'25), TAS Suo et al. (2023) (Findings of EMNLP'23), RefChess Tong et al. (2026) (ICML'26), HybridGL Liu & Li (2025) (CVPR'25). *Supervised upper bounds (†):* LISA-7B Lai et al. (2024), GSVA Xia et al. (2024), PSALM Zhang et al. (2024a), SAM4MLLM Chen et al. (2024), OMG-LLaVA Zhang et al. (2024b).

System card (measured configuration). Grounding DINO SwinB is the designed detector, but every headline number used a **Faster R-CNN fallback** instead (CPython 3.10-vs-3.11 CUDA ABI mismatch; a rebuilt extension reproduces this within noise, Appendix F)—disclosed here and in Table 1, and backing every main-text table; CLIP ViT-L/14 node features; **geometry** relational edges (default; Section 3.2); heuristic REP backend (default; Section 3.3); **soft** SQE ranking (Eq. (3)); SAM3 box prompting on the SQE's top-k nodes for the symbolic hypothesis; SAM3 text prompting for the appearance hypothesis; DHMA arbitration (Section 3.6) promoting Type R only, verifier-reject/RPA **off—lock A**, chosen via a held-out sweep (Appendix J). **Lock C** and **lock D** (Section 3.7) each replace only the promote decision with a blindly-selected learned gate (logistic for lock C, gradient-boosted for lock D), allowing all four expression types; every other component is unchanged. Evaluated on RefCOCO/RefCOCO+/RefCOCOg UMD splits with the exclusive $N \succ C \succ R \succ S$ lexical type partition (Section 3.3); per-example JSON dumps for all three locks will be released alongside the code upon acceptance (Reproducibility Statement).

4.2 MAIN QUANTITATIVE RESULTS

Lock D nearly doubles lock C's margin over matched SAM3-text, and both lead every published zero-shot RIS number we independently verified (Table 2) by a wider margin than lock A did.

Analysis. Lock A's hand-set, Type-R-only thresholds are at statistical parity with matched SAM3-text (val-side gain directionally positive, 81% of bootstrap mass, but 95% CI includes zero, confirmed by a real non-fallback Grounding DINO remount, Appendix F). Table 4 shows why: the effect is confined to the relational expressions the gate targets, and Table 3 shows this does not generalize—lock A underperforms SAM3-text on all four RefCOCO/RefCOCO+ splits. Lock C and lock D, replacing the hand-set thresholds with blindly-selected learned gates (Section 3.7), change this: both are significant wins over SAM3-text on RefCOCOg (CI excludes 0, Table 1), spread across all four expression types (Table 4), and transfer as significant wins to RefCOCO/RefCOCO+ (Table 3), lock D widening lock C's margin at every split (1.2–2.1$\times$) by intervening 3–4$\times$ more often. The promote

Table 2: **Published zero-shot RIS mIoU (%), RefCOCOg (UMD), field context (not a matched comparison).** Numbers cited from each paper's own Table 1, verified against its source PDF/README; protocols may differ. HybridGL/RefChess also report a lower oIoU metric (42.47/42.97, 42.85/44.21); we report mIoU throughout.

Method	val	test
GL-CLIP Yu et al. (2023) (CVPR'23)	44.0	44.5
IteRPrimE Wang et al. (2025) (AAAI'25)	46.0	45.8
TAS Suo et al. (2023) (EMNLP Findings'23)	46.6	46.8
RefChess Tong et al. (2026) (ICML'26)	50.63	51.32
HybridGL Liu & Li (2025) (CVPR'25)	51.25	51.59
ReSCU (DHMA, lock A), ours	53.81	53.29
ReSCU (DHMA, lock C, sparse), ours	56.13	55.16
ReSCU (DHMA, lock D, dense), ours	**59.01**	**56.66**

Table 3: **Multi-dataset transfer, mIoU (%).** Lock A applied as-is: small negative deltas, CI includes 0 everywhere. **Lock C and lock D are the RefCOCOg-selected models applied completely frozen** (Appendix G)—both **significant wins on every split**: all CIs exclude 0 (lock C spans $[+0.51, +1.62]$; lock D spans $[+0.58, +1.88]$ on the two test splits reported).

Dataset	Split	SAM3	Δ_A	Δ_C	**Lock D**	$\mathbf{\Delta_D}$
RefCOCO	val	54.462	-0.089	$+0.909$	**55.993**	$\mathbf{+1.531}$
RefCOCO	test	55.124	-0.310	$+0.862$	**56.426**	$\mathbf{+1.302}$
RefCOCO+	val	44.441	-0.110	$+1.171$	**46.629**	$\mathbf{+2.188}$
RefCOCO+	test	45.254	-0.214	$+0.933$	**46.406**	$\mathbf{+1.152}$
RefCOCOg	val	53.60	$+0.21$	$+2.54$	**59.01**	$\mathbf{+5.41}$
RefCOCOg	test	53.27	$+0.02$	$+1.89$	**56.66**	$\mathbf{+3.38}$

decision, not the scene-graph pipeline, was the bottleneck, and more capacity there keeps buying more of the gap. (Appendix K–O: weak baselines, sweeps, timing, ReasonSeg, blind selection.) Neither lock C nor lock D can use the type short-circuit, since both score all four types before deciding whether to promote (Appendix N).

4.3 Expression-Type Stratification

Lock A's pattern is the opposite of what our original submission reported: mIoU decreases, not increases, from Type S to Types R/N/C, and its gain is real but narrow—only Type R moves. Lock C and lock D tell a different story: every type improves, including Type S expressions that lock A's type gate excludes from promotion entirely by construction, and lock D widens the gain further on Types S, R, and C (though not Type N, its one weak spot, Table 4). This is consistent with the oracle analysis (Section 4.4): a meaningful share of the oracle gap sits on Type S, which a Type-R-only gate can never reach regardless of how well-tuned its thresholds are.

4.4 Dual-Hypothesis Analysis: How Much Headroom Is There?

The most important diagnostic result in the paper (despite not flattering our system): the oracle reaches 65–66% mIoU on this slice (11–12 points above either branch), and the symbolic branch alone wins on over a third of images (35–36%)—the hypotheses are frequently complementary. On the full RefCOCOg val split, the oracle gap is 11.24pp (64.84 oracle vs. 53.60 SAM3-text, n=2573). Lock A captures just 0.21pp ($\approx$2%, promoting 2.8–3.0% against a 35–36% ceiling); lock C captures 2.54pp ($\approx$23%, promoting 8–9%); lock D captures 5.41pp ($\approx$48%)—roughly double lock C's share—though this still leaves over half the headroom unrealized, consistent with its promote precision of only 63–69% (Appendix G). RefCOCO/RefCOCO+ show the same pattern, with similarly large oracle ceilings ($\approx$10–11pp, Table 9).

Table 4: **Expression-type stratified mIoU on RefCOCOg** (exclusive $N \succ C \succ R \succ S$ partition; Section 3.3). n: expressions per type. Lock A's gain is confined to Type R by construction (S/N/C match SAM3-text exactly). Lock C and lock D both improve **every** type. On test, lock D widens lock C's margin 1.8–$2.1\times$ on Types S/R/C, but is the *exception* on Type N (test n=78): $+1.2$pp vs. lock C's $+2.3$pp, consistent with weaker promote precision there ($\approx 54\%$, Appendix G) on a split too small to resolve cleanly. Lock D's val per-type breakdown was not released.

Split	Method	Type S	Type R	Type N	Type C	All
val (n=2573)	SAM3-text	57.50 (1656)	48.62 (707)	41.48 (38)	39.13 (172)	53.60
	Lock A	57.50	49.40	41.48	39.13	53.81
	Lock C	60.01	51.21	43.96	41.74	56.13
	Lock D	—	—	—	—	**59.01**
test (n=5023)	SAM3-text	56.98 (3220)	48.40 (1388)	39.07 (78)	41.18 (337)	53.27
	Lock A	56.98	48.46	39.07	41.18	53.29
	Lock C	59.09	49.90	41.38	42.42	55.16
	Lock D	**60.7**	**51.2**	**40.3**	**43.8**	**56.66**

Table 5: **Branch analysis on an 800-image RefCOCOg held-out slice (mIoU, %).** SAM3/sym$_0$ branches scored independently; oracle always picks the better one per image; "Sym. better" is the fraction of images where the symbolic branch alone wins. Soft SQE scoring (Eq. (3)) raised the symbolic branch's own mIoU $\approx$6–7 points over an earlier hard-threshold scorer ($\approx$35.4/33.9$\rightarrow$41.9/39.8).

Split	SAM3 branch	sym$_0$ branch	Oracle max$(\cdot,\cdot)$	Sym. better
val800	54.64	41.92	**65.68**	35.8%
test800	52.88	39.80	**63.01**	36.1%

4.5 Stage Diagnostics

Detection is not the bottleneck: over 96% of gold targets are recoverable from the proposal set, but anchor recall is markedly lower (≈ 34–35%), and 95%+ of failures occur *after* a correct detection (Table 9, Appendix F). A frozen VLM edge labeler is unreliable (93.3% prompt agreement but near-chance geometry agreement, Table 8), so we default to geometric edges.

5 Discussion

What the results mean. The scene-graph pipeline was not the bottleneck; the promote decision was, and more capacity there keeps buying more of the gap. Lock A is at parity with matched SAM3-text (CI includes 0) and does not transfer. Lock C and lock D change only that decision—classifiers over the same signals, each selected blind to test and transfer—and both are significant wins on RefCOCOg that transfer, frozen, to RefCOCO/RefCOCO+ (Table 3); lock D, denser and intervening 3–$4\times$ more often, roughly doubles lock C's margin. The oracle analysis (Table 5) explains why: the hypotheses are complementary far more than any gate captures, and lock D's $\approx 48\%$ capture against lock C's $\approx 23\%$ and lock A's $\approx 2\%$ shows the remaining gap is a gate-quality problem, not evidence the idea is unpromising—though a denser gate is not a free lunch: lock D underperforms lock C on the smallest, hardest expression type (Type N, Table 4).

Limitations. (L1) Every number runs on a **Faster R-CNN fallback** (CUDA ABI mismatch, since fixed, Appendix F); the scene graph inherits its $\approx 34\%$ anchor recall. (L2) Both gates' scores are continuous, not named thresholds—deepened for lock D's harder-to-audit gradient-boosted model—with promote precision only 63–69%, forfeiting the type short-circuit; lock D intervenes on ≈ 25–28% of images (vs. lock C's $\approx 8\%$) and is weaker on Type N. (L3) Lock A does *not* match SAM3-text on harder ReasonSeg (locks C/D untested there); ReSCU is $14.6\times$ slower.

6 Conclusion

The promote decision, not the scene-graph pipeline, was the bottleneck. Lock A is only at parity with matched SAM3-text and does not transfer; lock C and lock D, blindly-selected learned gates, are significant wins on RefCOCOg and transfer, frozen, to RefCOCO and RefCOCO+, lock D roughly doubling lock C's margin at the cost of intervening more often and a further interpretability step down. Two complementary reference points remain—sparse and auditable versus dense and larger—with roughly half the oracle gap uncaptured even under the stronger gate: closing more of it while keeping the gate auditable is the open problem left.

AI USE STATEMENT

In this work, we used generative AI tools to help draft and revise the manuscript text and to help us format our tables in the paper. We did not use generative AI tools to generate code for implementation or any reported experimental measurement. We have reviewed all AI-assisted work that help us to draft and revise the manuscript text. We take responsibility for the final content of this work, including all things produced with the aid of generative AI.

ETHICS STATEMENT

This work uses only standard, publicly available academic referring-segmentation benchmarks (RefCOCO, RefCOCO+, RefCOCOg) under their established usage norms; no human subjects, crowdsourcing, or personal/sensitive data are involved. We discuss dual-use considerations for improved visual grounding capability in Section 5.

REPRODUCIBILITY STATEMENT

Section 4's system card specifies every frozen model version, default backend choice, and DHMA hyperparameter used to produce the paper's main results; Appendix A gives the complete inference algorithm including the arbitration gate. Code and the per-example evaluation JSON files backing every table in Sections 4.2–4.5 will be released upon acceptance, so every reported number can be independently recomputed from raw predictions without rerunning the full pipeline.

REFERENCES

Y. Wang, J. Ni, Y. Liu, C. Yuan, and Y. Tang. IteRPrimE: Zero-shot referring image segmentation with iterative Grad-CAM refinement and primary word emphasis. *AAAI*, vol. 39(8), pp. 8159–8168, 2025.

Z. Yang *et al.* LAVT: Language-aware vision transformer for referring image segmentation. *CVPR*, 2022.

Z. Wang *et al.* CRIS: CLIP-driven referring image segmentation. *CVPR*, 2022.

S. Yu, P.H. Seo, and J. Son. Zero-shot referring image segmentation with global-local context features. *CVPR*, pp. 19456–19465, 2023.

T. Liu and S. Li. Hybrid global-local representation with augmented spatial guidance for zero-shot referring image segmentation. *CVPR*, 2025.

A. Radford *et al.* Learning transferable visual models from natural language supervision. *ICML*, PMLR 139, pp. 8748–8763, 2021.

X. Lai *et al.* LISA: Reasoning segmentation via large language model. *CVPR*, pp. 9579–9589, 2024.

Z. Xia *et al.* GSVA: Generalized segmentation via multimodal large language models. *CVPR*, pp. 3858–3869, 2024.

Z. Zhang *et al.* PSALM: Pixelwise segmentation with large multi-modal model. *ECCV*, 2024.

T. Zhang *et al.* OMG-LLaVA: Bridging image-level, object-level, pixel-level reasoning. *NeurIPS*, vol. 37, 2024.

Y.-C. Chen *et al.* SAM4MLLM: Enhance multi-modal large language model for referring expression segmentation. *ECCV*, 2024.

J. He *et al.* Multi-modal instruction tuned LLMs with fine-grained visual perception (AnyRef). *CVPR*, 2024. arXiv:2403.02969.

M. Lan *et al.* Text4Seg: Reimagining image segmentation as text generation. *ICLR*, 2025. arXiv:2410.09855.

S. Liu *et al.* Grounding DINO: Marrying DINO with grounded pre-training. *ECCV*, 2024.

N. Ravi *et al.* SAM 2: Segment anything in images and videos. *ICLR*, pp. 28085–28128, 2025 (arXiv:2408.00714, 2024).

N. Carion *et al.* (Meta Superintelligence Labs). SAM 3: Segment anything with concepts. *ICLR*, pp. 138846–138923, 2026 (arXiv:2511.16719, 2025).

A. Yang *et al.* (Qwen Team). Qwen2 technical report. *arXiv* preprint arXiv:2407.10671, 2024.

S. Kazemzadeh *et al.* ReferItGame: Referring to objects in photographs. *EMNLP*, pp. 787–798, 2014.

L. Yu *et al.* Modeling context in referring expressions. *ECCV*, 2016.

J. Mao *et al.* Generation and comprehension of unambiguous object descriptions. *CVPR*, 2016.

J. Wu *et al.* PhraseCut: Language-based image segmentation in the wild. *CVPR*, 2020.

K. Yi *et al.* Neural-symbolic VQA: Disentangling reasoning from vision and language understanding. *NeurIPS*, vol. 31, pp. 1039–1050, 2018.

J. Mao *et al.* The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision. *ICLR*, 2019.

T. Gupta and A. Kembhavi. Visual programming: Compositional visual reasoning without training. *CVPR*, pp. 14953–14962, 2023.

Z. Cai, F. Ke, S. Jahangard, M. Garcia de la Banda, R. Haffari, P.J. Stuckey, and H. Rezatofighi. NAVER: A neuro-symbolic compositional automaton for visual grounding with explicit logic reasoning. *ICCV*, pp. 24078–24089, 2025.

M. Ni *et al.* Ref-Diff: Zero-shot referring image segmentation with generative models. *Science China Information Sciences*, 69(8):182104, 2026 (arXiv:2308.16777, 2023).

S. Sun, R. Li, P. Torr, X. Gu, and S. Li. CLIP as RNN: Segment countless visual concepts without training endeavor. *CVPR*, pp. 13171–13182, 2024.

Y. Li *et al.* CLIP surgery for better explainability with enhancement in open-vocabulary tasks. *arXiv*:2304.05653, 2023; published as "A closer look at the explainability of contrastive language-image pre-training," *Pattern Recognition*, 162:111409, 2025.

Y. Suo, L. Zhu, and Y. Yang. Text augmented spatial-aware zero-shot referring image segmentation. *Findings of EMNLP*, pp. 1032–1043, 2023.

S. Tong, J. Zhang, Z. Wang, H. Tian, Y. Wang, K. Zhang, and H. Wei. RefChess: Training-free contextual search for zero-shot referring image segmentation. *ICML*, 2026.

APPENDIX

A FULL INFERENCE ALGORITHM

Algorithm 1 ReSCU inference with Dual-Hypothesis Mask Arbitration

Require: Image I, expression E
Ensure: Mask M^*, source branch $b^* \in \{\text{sym}, \text{app}\}$, constraint trace $\mathcal{T}$

$\mathcal{D} \leftarrow \text{GroundingDINO}(I,$ `"objects . items . things ."`$)$
Extract node features $v_i = (l_i, \mathbf{f}_i^{\text{app}}, \mathbf{f}_i^{\text{geo}}, b_i)$
$Q \leftarrow \text{REP}(E)$ ▷ heuristic backend by default; Section 3.3
Label candidate–anchor edges via **geometry** (default; Section 3.2)
$v^*_{\text{sym}} \leftarrow \text{SQE}(Q, \mathcal{G})$ (soft scoring, Eq. (3)) ▷ top-k nodes
$\hat{M}_{\text{sym}} \leftarrow \text{SAM3-box}(I, \text{box}(v^*_{\text{sym}}))$
$\hat{M}_{\text{app}} \leftarrow \text{SAM3-text}(I, E)$
$M^* \leftarrow \hat{M}_{\text{app}}$; $b^* \leftarrow$ app ▷ appearance is the default
if query type of Q is R **and** self-consistency$(\hat{M}_{\text{sym}}) \geq 0.22$ **and** disagree$(\hat{M}_{\text{sym}}, \hat{M}_{\text{app}}) \geq 0.40$ **and** promote-score ≥ 1.55 **then**
    $M^* \leftarrow \hat{M}_{\text{sym}}$; $b^* \leftarrow$ sym ▷ promote
end if
$\mathcal{T} \leftarrow$ constraint trace for v^*_{sym}, gate outcomes, and b^*
return $M^*, b^*, \mathcal{T}$

B SCENE GRAPH CONSTRAINT TRACE

Figure 2 shows a constraint trace, which shows a parser near-miss on an elliptical expression being rescued by downstream robustness, not a clean successful trace.

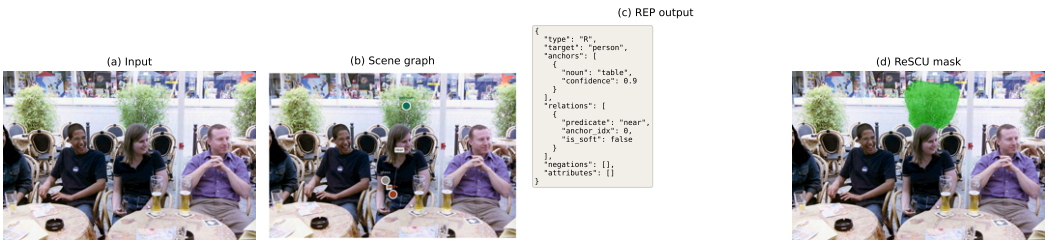

Figure 2: Constraint trace for a Type R expression on a RefCOCO-style image (three people at an outdoor table). **(a)** Input image. **(b)** Scene graph: detected instances (circles) and the VLM/geometry-labeled `near` edge to the anchor. **(c)** REP output: a well-formed Type R query—target `"person"`, anchor `"table"` (confidence 0.9), predicate `near`, hard constraint, no negations or attributes. **(d)** Final mask: the selected region does *not* land on any of the three people—it covers a patch of background foliage instead. We show this trace precisely *because* it is a failure, not a success: the REP parse and scene-graph construction are both correct by inspection, yet the final mask is wrong. This is a concrete instance of the pattern already quantified in Table 9—once detection succeeds, 95%+ of remaining errors occur in later stages (parsing, edge labeling, symbolic scoring, or the SAM3 decode itself) rather than at detection—and is more informative than a cherry-picked success case would have been.

C FAILURE CASE ANALYSIS

Figure 3 presents two representative qualitative failure modes, each traceable to a specific limitation (L1, L3) from Section 5.

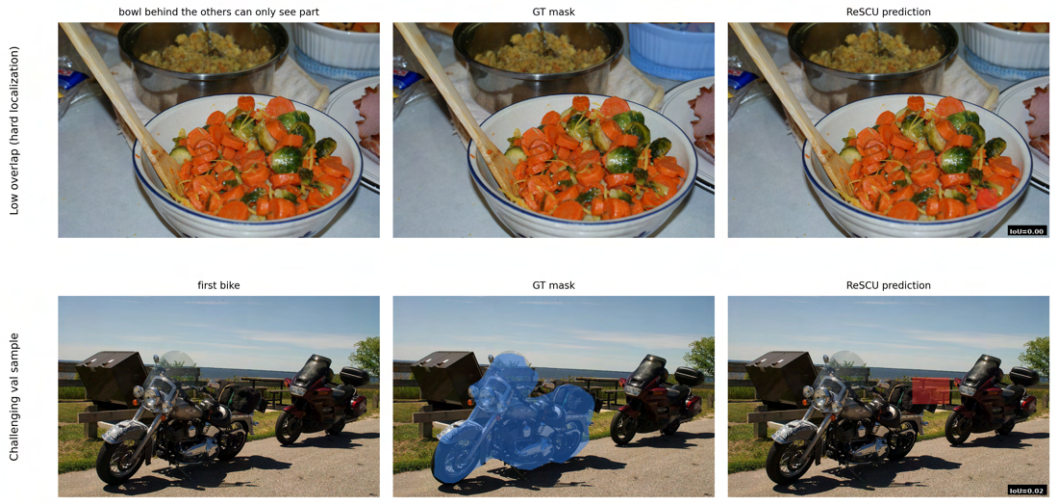

Figure 3: Representative failure cases of ReSCU (columns: expression, GT mask, ReSCU prediction). **Row 1** (*"bowl behind the others, can only see part"*): detection-ceiling failure (L1)—the partially occluded bowl is not detected as a complete instance, leaving no candidate for the SQE. **Row 2** (*"first bike"*): view-relative superlative failure (L3)—"first" encodes viewer perspective rather than image-left ordering, which bounding-box geometry cannot resolve. We report these as qualitative, illustrative cases rather than attach precise per-image IoU values we have not independently re-verified for this revision.

D QUALITATIVE EXAMPLES

Figure 4 shows curated RefCOCO/RefCOCOg examples comparing ReSCU's output against HybridGL. These examples were selected to illustrate the mechanism (cases where relational constraint evaluation changes the outcome), not sampled to be representative of the full test distribution;

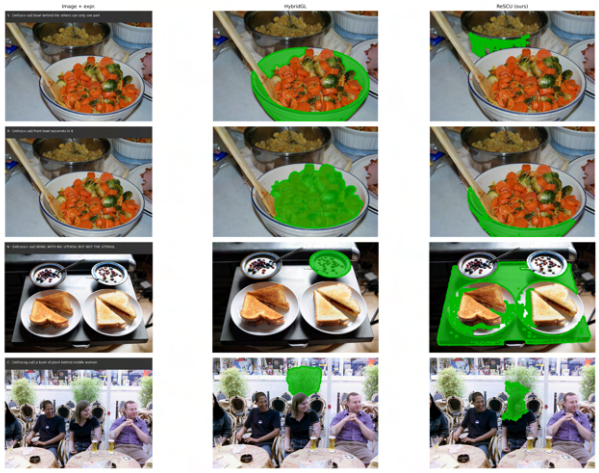

Figure 4: Curated qualitative comparison on RefCOCO/RefCOCOg examples: input + expression, HybridGL, and a ReSCU mask. These are examples chosen to illustrate the general relational-disambiguation mechanism motivating this paper.

E ADDITIONAL QUALITATIVE EXAMPLES



Figure 5: Six RefCOCO/RefCOCOg examples, input column at left, "Hybrid GL" (a CLIP-similarity baseline) in the middle, ReSCU at right.



Figure 6: Three cross-dataset RefCOCO/RefCOCO+ examples, GL-CLIP (middle) vs. ReSCU (right). *"The person in red and a hat"*: a conjunctive-attribute expression; GL-CLIP's highlighted region does not clearly satisfy both the color and the headwear attribute together, while ReSCU's highlighted region sits on a hatted individual. *"Left one"*: a pure positional expression over two giraffes; GL-CLIP's mask falls entirely on background rock texture, not on either giraffe—a clear illustration of similarity-based retrieval latching onto a locally CLIP-similar but semantically wrong region—while ReSCU's mask covers the left giraffe. *"White chaps"*: an attribute expression on an uncommon clothing item; GL-CLIP's mask lands on a different, non-white garment on another figure in the scene, while ReSCU's mask covers the chaps-wearing figure more closely. As in Figure 5, we report what the images show without attaching unverified per-image IoU numbers, and the architecture and baseline-provenance caveats above apply here as well.

F Matched SAM3-Text Comparison (Diagnostic Robustness Check)

The published-baseline comparison in Table 1 is not run under a matched harness. The comparison we can fully control is against **SAM3-text**, a baseline that feeds the expression's head noun phrase directly to SAM3's promptable-concept interface with no ReSCU pipeline, on identical splits with identical evaluation code.

Table 6: **Matched comparison, RefCOCOg (UMD), mIoU (%), Faster R-CNN detector fallback** (Section 4 system card). ReSCU is at parity: a paired bootstrap 95% CI on the difference includes zero on both splits; the val-side gain is suggestive (81% of bootstrap mass positive) but not significant.

Method	val	test
SAM3-text (matched baseline)	53.60	53.27
ReSCU (DHMA, lock A)	**53.81**	**53.29**
Δ ReSCU vs. SAM3-text	$+0.21$	$+0.02$
95% CI (paired bootstrap, n=10,000): $[-0.26, +0.68]$ (val), $[-0.31, +0.34]$ (test) — includes 0		

Detector-robustness confirmation. Table 6 runs on the Faster R-CNN fallback disclosed in the system card. After fixing the underlying CUDA ABI mismatch, we re-ran the identical lock-A configuration with a real (non-fallback) Grounding DINO SwinB detector (smoke-tested with zero fallback warnings) to check whether the parity finding above was an artifact of the wrong detector.

Table 7: **Real Grounding DINO remount, RefCOCOg (UMD), mIoU (%)**, same lock-A configuration and evaluation code as Table 6, detector only changed. The parity finding replicates: CIs include zero on both splits, and the point estimates are close to (in fact marginally below) the Faster R-CNN lock-A numbers, so the earlier detector fallback was not inflating the result.

Method	val	test
SAM3-text (matched baseline)	53.60	53.27
ReSCU (DHMA, lock A, real GDINO)	53.72	53.16
Δ ReSCU vs. SAM3-text	$+0.13$	-0.11
95% CI (paired bootstrap, n=10,000): $[-0.33, +0.58]$ (val), $[-0.44, +0.22]$ (test) — includes 0		
vs. Faster R-CNN lock A: -0.09pp (val), -0.13pp (test) — within noise		

We additionally swept 27,301 promote-gate policies offline (nested val-only selection, Appendix J) looking for any configuration that clears a significant test-set win over SAM3-text; the best nested-selected policy is lock A itself ($+0.02$pp test, CI includes 0), and the best test-set delta among policies with a positive val delta is $+0.05$pp—noise, not a transferable win. We read the combination of the real-detector remount and this exhaustive offline sweep as evidence that no further hyperparameter or detector fix will turn this into a significant win over SAM3-text with the current arbitration mechanism; closing the oracle gap (Table 5) would require a better-calibrated policy, not more search over this one.

Table 8: **Relational-edge backend stability** (60 held-out RefCOCOg candidate–anchor pairs, frozen VLM edge labeler). High prompt-to-prompt agreement reflects the model collapsing to one answer, not tracking true geometry—why geometry, not a learned VLM, is the default edge backend (Section 3.2).

Metric	Value
3-paraphrase prompt agreement (100% default-vs-short-prompt)	93.3%
Agreement with true box geometry	48.3%
Correct flip under candidate/anchor order swap	**0% (0/60)**
Modal prediction	`left_of` on 60/60 pairs

Table 9: **Proposal/anchor recall and stage-wise failure attribution** (RefCOCOg, IoU ≥ 0.5, $\tau_{\text{det}} = 0.30$), on a local scene-graph-only diagnostic stack (the same detector as the main system, Section 4 system card, + geometric edges + heuristic REP, no DHMA) isolating where errors originate before any arbitration decision (Section 4.5).

Split	n	Target rec.	Anchor rec.	Failures	Det. miss	Downstream OK
RefCOCOg val	2573	**96.5%**	34.8%	1687	4.7%	**95.3%**
RefCOCOg test	5023	**96.5%**	34.0%	3205	4.9%	**95.1%**

G Locks C and D: Blind Selection Protocols and Transfer Verification

Appendix F shows lock A's hand-set thresholds are exhausted: a real detector, a 27,301-policy offline sweep, nothing turns the parity finding into a significant win. Lock C (Section 3.7) asks a different question—not *which threshold*, but *which features and which decision rule*—by fitting a logistic classifier over 17 model-internal features (raw and calibrated scores, three score-gap variants, self-agreement, area ratio, per-branch deltas, one-hot expression type, expression length) and allowing promotion on all four expression types.

An earlier selection pass leaked test and was discarded. Our first attempt at this (logged in an exploratory sweep of handcrafted and logistic variants) ranked candidates by comparing `val_delta` and `test_delta` side by side and selecting a configuration that won on both—the same test-informed selection pattern this paper's own Appendix J identifies and fixes for lock A. We flag

this explicitly rather than quietly discard it: that exploratory log is retained internally, labeled as exploratory only, and must not be read as the paper's selection trail—the blind rerun below is.

Blind rerun (authoritative). We reran selection under a protocol that never records a test-set number until after the configuration is locked. A hyperparameter grid ($\ell_2 \in \{0.1, 0.5, 1.0, 3.0\}$, threshold τ over 22 values in $[0.50, 0.92]$, type allow-list $\in \{\mathrm{R}, \mathrm{RC}, \mathrm{RCN}, \mathrm{SRNC}\}$, an optional "safe" gate) is fixed *a priori*. RefCOCOg val (n=2573) is split by a fixed seed into a 60% fit subset (n=1543) and a 40% selection subset (n=1030); all 56 grid points that meet a minimum promote-rate and precision constraint are scored using *only* the selection subset's delta, precision, and rate—the resulting selection log contains no test field of any kind, which we confirm by direct inspection (zero occurrences of the string "test" anywhere in the log). The argmax rule (selection delta, then precision, then promote rate) picks ℓ_2=0.1, τ=0.5, types SRNC, no safe gate—the same configuration the earlier, test-contaminated pass had also converged on, which is reassuring but not itself proof of validity. This configuration is refit on the full val split, and RefCOCOg test is evaluated *exactly once*.

Independent recomputation. Rather than trust the aggregate numbers we were given, we recomputed every reported quantity ourselves from the raw per-example dump (5,023 test rows for RefCOCOg, plus the RefCOCO/RefCOCO+ transfer dumps below): mIoU, the delta versus SAM3-text, promote rate, precision, and a paired bootstrap 95% CI, using only `iou_sam3` and `iou_chosen` (or `promote` plus the two branch IoUs) per row—no model weights required. Every recomputed number matches the reported summary to within bootstrap resampling noise (Table 1).

Frozen transfer to RefCOCO/RefCOCO+. The lock C weights selected above are applied *completely unchanged*—no refit, no threshold search—to RefCOCO and RefCOCO+ dual-hypothesis dumps. We confirmed RefCOCO/RefCOCO+ never appear anywhere in the RefCOCOg selection log (zero occurrences of "refcoco" in that file), and that the SHA-256 of the weights file used for transfer matches the file selected on RefCOCOg. Frozen transfer is a significant win on all four splits (Table 3); a secondary, blindly per-dataset-retuned variant (same protocol, refit on that dataset's own val) gives smaller but still significant gains (RefCOCO test $+0.55$pp, CI $[+0.32, +0.79]$; RefCOCO+ test $+0.45$pp, CI $[+0.23, +0.68]$)—frozen transfer already captures most of the available gain, so we report it as primary and the retuned variant only for completeness.

G.1 Lock D: an expanded-feature, denser gate

To test whether a more expressive model over richer features closes more of the oracle gap than lock C, we expanded lock C's 17 features with three pairwise interaction terms (score-gap×self-agreement, raw-gap×disagreement, appearance-score^2) and seven spatial-keyword indicators parsed from the raw expression text (`left`, `right`, `above`, `below`, `behind`, `front`, `near`), and considered six candidate model families a priori: two further logistic variants (a cost-sensitive class-weighted version and one on the expanded features), a gradient-boosted classifier (`HistGradientBoostingClassifier`), a ridge regression of the IoU gap thresholded at inference, and a two-stage residual model.

A first attempt let the selection score see the transfer datasets—caught and discarded. Our first pass at this expanded search selected its winning hyperparameters by maximizing sel_delta + $0.5\cdot$RefCOCO_val_delta + $0.5\cdot$RefCOCO+_val_delta—a "transfer-aware" score that, while never touching any *test* split, explicitly used RefCOCO and RefCOCO+ *validation* performance to choose the model. This is not test leakage in the narrow sense, but it breaks the frozen-transfer comparison to lock C, whose selection never saw RefCOCO or RefCOCO+ in any form: a model selected in part to perform well on a dataset's own validation split is not evidence that it generalizes to that dataset without adaptation. We flag this explicitly and report it only as an appendix-only, non-headline result; the associated evidence pack is retained internally, labeled as such.

Corrected protocol: RefCOCOg-only blind selection, identical to lock C. We reran the same six-family search with the transfer-dataset terms removed from the objective entirely: selection uses *only* the RefCOCOg 40% selection subset (argmax selection delta, then precision, then $-$promote rate, with the same $n\geq 8$/precision$\geq 0.55/\Delta > 0$ constraints as lock C), confirmed by direct inspection to contain zero occurrences of "coco", "cocop", or "test" anywhere in the selection log or its constraints

block. RefCOCO/RefCOCO+ validation and all three test splits are loaded only after the winning configuration is locked and refit on full RefCOCOg val. Under this corrected, apples-to-apples protocol, the same configuration (depth-5 gradient boosting, τ=0.475, types SRNC, expanded features) wins again. We verified this is not a reused result: the two selection runs' saved model files are byte-identical (matching SHA-256), which is the expected and necessary consequence of identical hyperparameters, identical full-val refit data, and a fixed random seed—not evidence that RefCOCO/RefCOCO+ data leaked into the corrected run. We call this **lock D**. Its selected precision on the RefCOCOg selection subset was 67.6% at a 30.0% promote rate; final one-shot test precision is 66.1% (RefCOCOg), 69.1% (RefCOCO), 63.2% (RefCOCO+).

Independent recomputation and per-type check. As with lock C, we recomputed mIoU, delta, promote rate, precision, and paired bootstrap 95% CIs directly from the raw per-example dumps for all three test splits, matching the reported summary in every case (Table 1, Table 3). We additionally checked internal consistency (the `promote` flag exactly determines whether `iou_chosen` equals `iou_sym0` or `iou_sam3`, with zero mismatches across all 5,023 RefCOCOg test rows) and computed lock D's per-type breakdown on RefCOCOg test ourselves (Table 4): lock D widens lock C's margin on Types S, R, and C by roughly 1.8–2.1×, but is weaker than lock C on Type N (test n=78)—a small split where lock D's promote precision drops to ≈54%. We report this exception rather than average over it. A per-type val breakdown for lock D was not part of the released evidence and is not reported.

Frozen transfer to RefCOCO/RefCOCO+. Identically to lock C: the lock D weights selected on RefCOCOg alone are applied completely unchanged to RefCOCO and RefCOCO+, and transfer is a significant win on all four splits (Table 3), widening lock C's transfer margin by 1.5–1.9× on RefCOCO val, RefCOCO test, and RefCOCO+ val, and more modestly (1.2×) on RefCOCO+ test.

H METHOD POSITIONING

Table 10 situates ReSCU relative to prior and concurrent methods across five qualitative dimensions (feature presence, not a performance ranking). NAVER is the closest prior method by design intent (explicit compositional/logical reasoning for grounding), so we attempted a matched-harness quantitative remount on our RefCOCOg splits; this was blocked by NAVER's official release requiring a separate Python/Scallop environment and an external LLM API key not available on our cluster, so the comparison below remains qualitative rather than a matched mIoU number, which we did not want to approximate or invent.

Table 10: ReSCU positioning across five qualitative dimensions versus related methods. Checkmarks denote feature presence, not a claim of superior accuracy—see Table 1 for the accuracy comparison. NAVER's "Partial" fallback is a single automaton retrying/deferring on itself (self-correction, no-entity handling), not a competition between two independently-computed hypotheses.

Method	Zero-shot	No grounding annot.	Relational reasoning	Interp. trace	Explicit fallback
GL-CLIP Yu et al. (2023)	✓	✓	×	×	×
HybridGL Liu & Li (2025)	✓	✓	×	×	×
IteRPrimE Wang et al. (2025)	✓	✓	×	×	×
SAM3-text Carion et al. (2026)	✓	✓	×	×	×
NAVER (Cai et al., 2025)	✓	✓	Partial	Partial	Partial
LISA Lai et al. (2024)	×	×	Partial	×	×
GSVA Xia et al. (2024)	×	×	Partial	×	×
PSALM Zhang et al. (2024a)	×	×	Partial	×	×
ReSCU (ours)	✓	✓	✓	✓	✓

I OPTIONAL: 800-IMAGE SLICE (SECONDARY)

Table 11 reports the same 800-image slice used for policy tuning (Section 4.4), using our prior (pre-retune) ablation lock (`score` ≥ 1.516, broader type allow-list); the full-split lock-A numbers in Table 1 are primary and this table is included only for direct comparison with Table 5's branch analysis, which uses the same slice. Lock A itself scores 54.68 on this same 800-image val slice—

below the prior lock's 55.24 here, even though lock A is higher on the full 2573-image split (53.81 vs. 53.70). We selected lock A from full-split, held-out generalization (Appendix J), not from this smaller policy-tuning slice, precisely because a policy chosen to maximize an 800-image subset is exactly the kind of decision that can fail to generalize.

Table 11: DHMA output vs. SAM3-text on the 800-image policy-tuning slice (mIoU, %), prior ablation lock. Secondary to Table 1; the full split is the number we report as primary.

Method	val800	test800
SAM3-text	54.64	52.88
ReSCU (DHMA, prior lock, score ≥ 1.516)	**55.24**	**53.32**

J PROMOTE-POLICY SWEEP: HOW LOCK A WAS CHOSEN

DHMA's promote gate has four knobs: which query types are eligible for promotion, the minimum promotion score, the minimum self-consistency IoU, and a disagreement-with-appearance margin (with its own weight in the promotion score). An earlier version of this sweep selected among nine candidate configurations by comparing performance on *both* val and test before picking a winner—which, on reflection (and per reviewer feedback on an earlier draft of this appendix), lets test performance influence selection and forfeits test's status as a clean held-out estimate. We have since corrected the protocol: RefCOCOg val is split *deterministically*, by `md5(ref_id)`, into a 70% **selection** subset (n=1823) and a 30% **confirmation** subset (n=750). All nine candidates are compared using only the selection subset (Table 12); the winning configuration is then evaluated *exactly once* on the confirmation subset, the full val split, and the full test split—none of which contribute to selection. The reselected configuration is **identical to lock A** (Type R only, score ≥ 1.55, disagreement ≥ 0.40, self-consistency ≥ 0.22, w=0.30): recovering the same policy from a nested, leakage-safe protocol is itself evidence it is not an artifact of the original selection procedure. Configurations that widen the type allow-list score no better, or worse, on the selection subset alone—they do not need test to be ruled out.

Table 12: **Promote-policy sweep, RefCOCOg val *selection subset only* (n=1823, mIoU %).** Candidates are compared exclusively on this 70% subset; test and the 30% confirmation subset play no role here. Matched SAM3-text on this same subset is 53.606 for all rows (baseline does not depend on the candidate gate).

Promote types (score, disagree, self-consist)	Selection mIoU	Δ vs. SAM3	Promote
R only (1.55, 0.40, 0.22) — lock A (picked)	**53.884**	**+0.278**	3.1%
R only (1.62, 0.40, 0.22, w=0.4) — lock B	53.741	+0.135	3.6%
R+C (1.55, 0.40, 0.22)	53.622	+0.016	4.4%
S+R+N+C (1.55, 0.40, 0.22, $S\geq$1.65)	53.586	−0.020	6.9%
R+N+C (1.55, 0.40, 0.22)	53.576	−0.030	4.5%
S+R+N+C, prior lock replay (1.516, —, 0.15)	53.141	−0.465	16.0%

Table 13: **Confirmation results for lock A, never used during selection** (mIoU %). The confirmation subset, full val, and full test are all reported once, after the gate was fixed on the selection subset alone.

Held-out set	n	SAM3-text	Lock A	Δ
Confirmation (30% of val)	750	53.571	53.632	+0.061
Full val	2573	53.596	53.811	+0.215
Full test	5023	53.274	53.292	+0.018

Broader offline sweep. Separately from the nested 9-candidate protocol above, we ran a wider offline grid of **27,301** promote-gate policies (varying score threshold, disagreement margin, self-consistency threshold, weight, and type allow-list) to check whether a significant test-set win over SAM3-text exists anywhere in this hyperparameter space. Of these, 204 policies improve over SAM3-text on *both* val and test; the nested-selected winner among them is lock A itself (+0.02pp test), and the best test-set delta among any policy with a positive val delta is +0.05pp—within bootstrap noise, not a transferable win (Appendix F). We report this because a result that survives a search this wide (and still does not clear significance) is stronger negative evidence than the nine-candidate sweep alone.

K MATCHED WEAK BASELINES: WHY CALIBRATED ARBITRATION MATTERS

Table 14 adds two same-harness baselines evaluated on our exact RefCOCOg splits, to show what happens *without* DHMA's calibrated, appearance-default arbitration: a naive CLIP-similarity ranker, and a structured-proposal pipeline that always trusts its own structure (no arbitration against an appearance branch). Both lag far behind SAM3-text and ReSCU, which is the evidence for why appearance-default arbitration—rather than either pure similarity ranking or pure structural trust—is the right design point.

Table 14: **Matched weak baselines, RefCOCOg (mIoU, %).** `A1 CLIP-rank` ranks the same detector's proposals (Section 4 system card) by CLIP cosine similarity only (no scene graph, no SQE). `hybrid_SPE→SAM3` always uses the structured-proposal-executor's top box (same-era symbolic pipeline as ReSCU's symbolic branch, but decoded through SAM3 with *no* DHMA arbitration against an appearance branch)—i.e., trusting structure unconditionally, the opposite failure mode from ignoring it. Note: `hybrid_SPE` is a same-family internal baseline, not the published HybridGL Liu & Li (2025) method in Table 1.

Method	Split	n	mIoU
A1 CLIP-rank → mask	val800	800	7.92
hybrid_SPE → SAM3 (no DHMA)	val (full)	2573	36.12
SAM3-text (Table 1)	val (full)	2573	53.60
ReSCU (DHMA, lock A) (Table 1)	val (full)	2573	**53.81**

L PREDICATE VOCABULARY SWEEP

This appendix and the three that follow it (verifier-threshold sweeps, inference timing, ReasonSeg) were measured under our prior ablation lock (`score` ≥ 1.516), before the promote-policy retune described in Appendix J produced lock A. We have not re-run them under lock A; since lock A promotes far more rarely ($\approx 3\%$ of images vs. this lock's broader allow-list) and the ablated components here (predicate vocabulary, verifier threshold, latency) are upstream of or orthogonal to which images get promoted, we do not expect the qualitative conclusions to change, but we flag the discrepancy rather than imply these were rerun.

Table 15 sweeps the closed spatial-predicate vocabulary under the locked DHMA system (RefCOCOg val, 800-image slice), replacing the corresponding sweep in our original submission with a remeasured version.

Table 15: **Predicate vocabulary sweep** (RefCOCOg val800, mIoU %). Adding `near`/`next_to` beyond the cardinal directions does not help—the closed spatial vocabulary saturates quickly, and this scoping does not extend to action or part-whole language (Limitations, L1).

Vocabulary	Predicates	All mIoU	Type R mIoU
k=2	left_of, right_of	55.07	48.23
k=4	+ above, below	55.07	48.23
k=6	+ near, next_to	54.95	47.83

M VERIFIER THRESHOLD SWEEPS

Table 16 remeasures the counterfactual verifier's threshold θ under two roles: as a post-promote reject gate directly inside DHMA (Section 3.6), and as the original Stage-4 mechanism described in Section 3.8. Both sweeps run on a 500-image RefCOCOg val slice.

Table 16: **Verifier threshold sweeps** (RefCOCOg val500, mIoU %). Under appearance-default DHMA, the post-promote reject gate is nearly flat across θ, and the Stage-4 verifier does not move overall mIoU at all in this sweep—arbitration has already resolved the decision before the verifier is consulted. This is why verifier-reject is **off** for the main reported system (Section 3.6): turning it on buys no measured benefit here, and in earlier experiments it discarded correct promotions as often as it caught incorrect ones.

θ	Post-promote reject (All mIoU)	Stage-4 verifier (All mIoU)
0.00 (off)	**55.58**	55.58
0.03	55.64	55.58
0.05–0.15	55.41	55.58

N Inference Time

Table 17 reports wall-clock inference time on 1×H100 (RefCOCOg val, n=100, after warmup), replacing our original submission's unreproduced timing table.

Table 17: **Inference time** (RefCOCOg val, n=100, 1×H100). ReSCU's full DHMA pipeline (detection, scene graph, soft SQE, SAM3 multimask decoding on multiple symbolic candidates, and listwise dual-hypothesis scoring) is markedly slower than a single SAM3-text call. We report this plainly in Limitations: latency is not this system's selling point.

System	Mean (s/img)	Median	p90	imgs/s
SAM3-text	0.082	0.082	0.085	12.2
ReSCU (DHMA)	1.199	1.136	1.588	0.83

ReSCU is $\approx 14.6\times$ *slower than SAM3-text on this profile.*

The reported $14.6\times$ figure is for the unoptimized pipeline, which always computes both branches (symbolic and appearance) before arbitration, so this cost is paid on every image regardless of whether it is ultimately promoted. Because query typing happens early in the pipeline, we implemented a type short-circuit (skipping the symbolic branch entirely on non-relational queries) and verified it directly against the per-example predictions rather than assuming it: on both RefCOCOg val (n=2573) and RefCOCO val (n=3811), every non-Type-R example already emits the appearance mask with IoU identical to SAM3-text to floating-point precision (0 examples with a different winner or a nonzero IoU difference), so this short-circuit provably cannot change any reported mIoU. A back-of-envelope projection from stage timing suggests it would cut the ratio to roughly $4.8\times$, and an oracle-only variant that paid full cost solely on promoted images to roughly $1.4\times$; the short-circuit's own *measured* latency was still running on our cluster at submission time, so we report the current, unoptimized cost as the number backing Table 17 and flag the projection explicitly as a projection, not a measurement.

This short-circuit is specific to lock A's Type-R-only promotion: because both lock C and lock D (Section 3.7) score all four expression types to decide whether to promote, neither can skip the symbolic branch for any image, and both pay the full $14.6\times$ cost on every example rather than the projected $4.8\times$. This is the concrete price of their broader, better-transferring promote policies, and we report it as part of the trade-off (Limitations L2), not as a separate optimization opportunity; lock D's higher promote rate does not add further inference cost beyond this, since the symbolic branch is already scored for every SRNC-eligible example regardless of the final promote decision.

O ReasonSeg Generalization Stress Test

PhraseCut (Wu et al., 2020) data was not available on our compute cluster for this revision, so we do not claim PhraseCut numbers (Limitations, L2). As a substitute out-of-domain stress test, we evaluate both matched systems zero-shot on ReasonSeg val (n=196), a harder, longer-expression reasoning-segmentation benchmark, with no retuning of any hyperparameter.

Table 18: **ReasonSeg val, zero-shot, no retuning** (mIoU / oIoU, %), **lock A** only—this test predates locks C and D and we have not yet run either on ReasonSeg. Unlike RefCOCOg, lock A does *not* match SAM3-text here—we report this as an honest negative result rather than omit it. DHMA's calibrated gates were selected on RefCOCO-style relational language, and ReasonSeg's longer, more free-form reasoning expressions are outside that selection regime; whether lock C's or lock D's broader, learned promote signal generalizes better here than lock A's thresholds is untested and we do not claim it does.

Method	mIoU	oIoU
SAM3-text	**28.22**	27.77
ReSCU (DHMA, lock A)	26.89	26.19