THE SCREEN IS NOT THE STATE: CAUSALLY GROUNDED ARTIFACT CRITICS FOR OFFICE AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Office agents produce editable documents, spreadsheets, and presentations whose correctness depends on more than visible content. Required formulas, heading styles, or speaker notes may be absent even when a file appears correct. Such failures can violate one requirement while leaving others satisfied and different implementations may satisfy the same requirements. Reliable evaluation therefore requires both access to native file state and an understanding of which rubric judgments should change or remain stable after an edit. We introduce CARE, a rubric-conditioned artifact critic that combines native and rendered evidence with programmatically verified relation supervision. Its **locality** objective encourages sensitivity on affected rubrics and consistency on unaffected rubrics and **invariance** stabilizes predictions across reward-preserving implementations. Inference requires no golden artifact. Experiments on a verified DOCX, XLSX, and PPTX corpus show reduced drift across three held-out splits. In an ablation study, adding locality to pairwise reward modeling reduces unaffected-rubric drift from 0.226 to 0.044, and adding invariance reduces reward-preserving candidate-score drift from 0.096 to 0.031. These results show that relational supervision can improve response stability without necessarily improving classification or candidate selection.

1 INTRODUCTION

Evaluating an office agent requires judging whether its delivered document, spreadsheet, or presentation satisfies user requirements (Ma et al., 2024; Ofengenden et al., 2025; Zhou et al., 2026). Appearance alone is insufficient: task-relevant native properties may be absent from rendered views. A heading can look correct without the required native style, or a spreadsheet total can display correctly without a required formula. Such failures can violate one requirement while leaving others satisfied. Consistent with constraint-level instruction-following evaluation (Jiang et al., 2024; Qin et al., 2024; Wen et al., 2024), artifact critics should assess individual task criteria, or **rubrics**, using requirement-relevant visible content and native state.

Computer-use benchmarks assess agents in operating-system and application workflows, including enterprise tasks (Xie et al., 2024; Rawles et al., 2025; Drouin et al., 2024; Boisvert et al., 2024). Visual reward estimation studies task-completion judgments from observations and trajectories (Yang et al., 2025; Lin et al., 2025), while office-specific evaluation can inspect saved deliverables through structural checks (Zhou et al., 2026; Ofengenden et al., 2025). Rendered evidence cannot distinguish **artifact aliases**: under the same instruction and rubrics, distinct artifact states can yield identical supplied visual evidence but different rubric rewards. A critic restricted to this evidence cannot resolve reward-relevant differences. Native-state access can expose these differences but doesn't ensure appropriate rubric-level judgments.

With native state observable, a critic must learn how rubric judgments should respond to artifact edits. As Figure 1 illustrates, edits changing some rubric outcomes should change corresponding judgments while leaving unaffected judgments stable. Implementation changes preserving every rubric outcome should leave all predicted rubric scores stable. Reward models and evaluators provide multi-dimensional or rubric-conditioned scores (Wang et al., 2024; Kim et al., 2024a;b; Anugraha et al., 2026), but output granularity alone doesn't specify how predictions should relate

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.

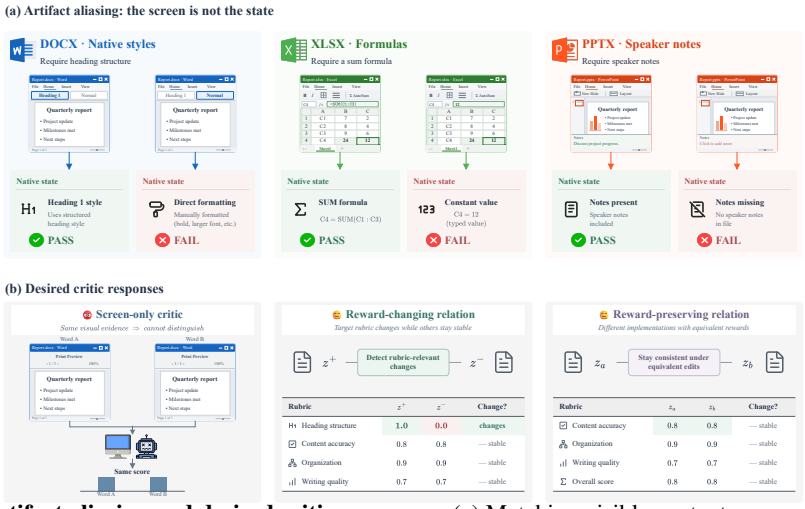


Figure 1: Artifact aliasing and desired critic responses. (a) Matching visible content can conceal different native states and rubric rewards in DOCX, XLSX, and PPTX. (b) A critic should detect rubric-relevant changes while remaining consistent on unaffected rubrics and across reward-equivalent implementations.

across candidates. Pointwise losses supervise candidates individually rather than directly comparing predictions, while aggregate preference learning constrains candidate order (Christiano et al., 2017) without specifying which rubric scores explain the difference. We ask: **with evidence access fixed across methods, can explicit relation supervision teach selective scoring responses?**

We introduce CARE, a rubric-conditioned artifact critic that learns **selective scoring responses** from controlled artifact relations with programmatically verified rubric outcomes. A deterministic selector assembles rubric-specific evidence from native state, initial-to-candidate changes, rendered views, and a global preservation summary. For each candidate-rubric pair, the critic predicts **fail**, **partial**, and **pass** probabilities with an evidence location; expected rubric rewards aggregate into a candidate score. Training augments pointwise supervision and candidate ranking with two relational objectives. On reward-changing pairs, **locality** encourages affected rubric scores to change in the verified direction while keeping unaffected predictions consistent. On reward-preserving pairs, **invariance** penalizes rubric-distribution and aggregate-score drift across equivalent implementations.

Our evaluation separates evidence access from response learning. Exploratory open-model controls compare render-only and render-plus-structure inputs. Adding locality to Artifact Pairwise-RM reduces unaffected rubric-score drift across all three test splits while retaining positive affected-rubric margins. Invariance further reduces candidate-score drift on reward-preserving pairs but lowers status macro-F1 on the source and synthetic executor-style splits in this configuration. On a fixed OfficeVal (Zhou et al., 2026) Best-of-4 pool, full CARE achieves a mean official score of 0.06705, versus 0.06999 without invariance and 0.05663 for random selection. The full objective improves reward-preserving stability but lowers the mean selection score relative to the no-invariance variant in this pool, alongside the classification trade-off.

Our contributions are as follows:

1. We formalize artifact aliasing and define reward-changing and reward-preserving relations to specify rubric-level sensitivity and stability.
2. We introduce CARE, a rubric-conditioned artifact critic learns selective scoring responses through locality and invariance objectives, without requiring a golden artifact at inference.
3. We construct a programmatically verified corpus spanning DOCX, XLSX, and PPTX, distinguish evidence access from response learning through controlled comparisons, and evaluate fixed-pool Best-of-4 selection on OfficeVal (Zhou et al., 2026).

2 Related Work

Computer-use and artifact evaluation. Computer-use benchmarks cover web, desktop, and mobile interaction, as well as enterprise workflows and compositional planning (Zhou et al., 2024; Xie et al., 2024; Rawles et al., 2025; Liu et al., 2024; Drouin et al., 2024; Boisvert et al., 2024). ZeroGUI uses VLM-based rewards for online GUI learning (Yang et al., 2025), while CUARewardBench evaluates computer-use reward models and combines prompt-level judgments through Unanimous Prompt Ensemble (UPE) (Lin et al., 2025). OfficeVal and PPTArena evaluate saved Office artifacts through executable, structural, and visual checks (Zhou et al., 2026; Ofengenden et al., 2025). SpreadsheetBench tests spreadsheet manipulation across multiple files per instruction (Ma et al., 2024), while PPTAgent assesses presentation content, design, and coherence (Zheng et al., 2025). LayoutLMv3 and the DocOwl family model document content and visual structure, including multi-page inputs (Huang et al., 2022; Hu et al., 2024; 2025). Complementing these approaches, CARE combines rendered content with native Office properties, such as formulas, fields, and relationships, for rubric-conditioned artifact evaluation without a golden artifact at inference.

Reward learning and structured judgments. Preference-based reward learning supports policy optimization and candidate selection (Christiano et al., 2017; Stiennon et al., 2020; Ouyang et al., 2022), while outcome and process supervision distinguish final-answer judgments from intermediate-step feedback (Uesato et al., 2022; Lightman et al., 2024). Similar provides step-wise, multi-dimensional rewards for virtual-agent actions (Miao et al., 2025). Structured evaluation includes explicit score rubrics in Prometheus and Prometheus 2 (Kim et al., 2024a;b), context-dependent aggregation of rating objectives in ArmoRM (Wang et al., 2024), and rubric-conditioned scores and explanations in R3 (Anugraha et al., 2026). For artifact outcomes, CARE augments pointwise judgment and ranking with rubric-level response constraints; aggregate ordering doesn't determine which rubric predictions should change across related candidates.

Counterfactual data and consistency learning. Counterfactually augmented data and contrast sets use small, meaningful edits to probe label-relevant distinctions (Kaushik et al., 2020; Gardner et al., 2020). CheckList distinguishes invariance from directional behavioral tests (Ribeiro et al., 2020), while consistency training encourages stable predictions under suitable transformations (Xie et al., 2020). RM-Bench separates subtle content differences from style bias (Liu et al., 2025). Closely related to our invariance objective, reWordBench tests meaning- or ranking-preserving transformations and trains score consistency across paraphrases (Wu et al., 2025). CARE builds on these ideas using programmatically verified native-state edits. It supervises affected-rubric sensitivity and unaffected-rubric consistency within reward-changing pairs, and constrains predictions across implementations whose complete rubric reward vectors remain unchanged.

3 Preliminaries

3.1 Task formulation

Let z_0 and z denote initial and candidate Office artifacts. Write $o = g(z, u)$ for the rendered observation under application state u. For instruction q and rubrics $C = \{c_i\}_{i=1}^K$, let $y_i(z) \in \{0, \frac{1}{2}, 1\}$ denote reward for rubric c_i in fail, partial, or pass. With task context (q, C, z_0) fixed, we define

$$\boldsymbol{y}(z) = \big(y_1(z), \ldots, y_K(z)\big), \qquad R(q, C, z) = \frac{\sum_{i=1}^K w_i y_i(z)}{\sum_{i=1}^K w_i}, \tag{1}$$

where $w_i \geq 0$ are rubric weights with $\sum_i w_i > 0$. The task dependence of $y_i(z)$ and $\boldsymbol{y}(z)$ is implicit. The target-free critic estimates these rewards from the instruction, rubrics, and evidence from the initial and candidate files. No golden artifact is required at inference; verifier outputs, relation labels, and private generation metadata are also excluded.

3.2 Artifact aliasing and response relations

Candidate pairs share the same task context (q, C, z_0). A pair (z_a, z_b) is **reward-changing** when $\boldsymbol{y}(z_a) \neq \boldsymbol{y}(z_b)$. Its affected and unaffected rubric sets are

$$A = \{i \in \{1, \ldots, K\} : y_i(z_a) \neq y_i(z_b)\}, \qquad U = \{1, \ldots, K\} \setminus A.$$

These sets are determined by rubric outcomes rather than edit locations.

A reward-changing pair is an **artifact alias** when its supplied rendered observations are identical:

$$z_a \neq z_b, \quad g(z_a, u_a) = g(z_b, u_b), \quad \boldsymbol{y}(z_a) \neq \boldsymbol{y}(z_b). \tag{2}$$

This equality concerns the supplied observations, not every possible application view. A critic restricted to the shared task context and these identical visual inputs cannot resolve the differing rubric outcomes. In our dataset, strict visual equality is checked under a frozen rendering pipeline by matching page counts, dimensions, RGB pixels, and normalized OCR.

A **reward-preserving pair** consists of distinct implementations with $\boldsymbol{y}(z_a) = \boldsymbol{y}(z_b)$. This requires equality of the complete rubric reward vector, not merely the aggregate reward; visual equality is not required. Reward-changing relations specify sensitivity on A and consistency on U, while reward-preserving relations specify stability across equivalent implementations.

4 METHOD

CARE learns a rubric-conditioned scoring function from artifact evidence and verified relations as shown in Figure 2. It combines a deterministic rubric-conditioned evidence selector, rubric-level status and evidence predictions, and relational objectives that constrain how predictions respond across related candidates. Relation labels, affected-rubric masks, verifier outputs, and candidate roles are used only for supervision and are never provided as model inputs.

4.1 ARTIFACT EVIDENCE AND CRITIC

Office adapters parse files according to the Office Open XML specification (Ecma International, 2016) into normalized native state and initial-to-candidate changes. For rubric c_i, a shared deterministic, non-trainable selector constructs evidence $E_i(z_0, z)$ from relevant native entities and dependencies, normalized diff events, rendered views or local crops, and a bounded global preservation summary. Native state exposes properties such as formulas, styles, fields, notes, and relationships that may be absent from a render, while the global summary records collateral changes outside the rubric's local target. The selector does not use verifier outputs, relation labels, mutation names, or candidate roles.

A multimodal backbone processes one candidate-rubric pair per forward pass. A status head predicts $\boldsymbol{p}_i \in \Delta^2$ over $\{\textit{fail}, \textit{partial}, \textit{pass}\}$, and an evidence head predicts one of the supplied evidence slots or `none`. Rubric and candidate scores are

$$s_i(z) = p_i(\text{pass}) + \tfrac{1}{2} p_i(\text{partial}), \qquad S(z) = \frac{\sum_{i=1}^{K} w_i s_i(z)}{\sum_{i=1}^{K} w_i}. \tag{3}$$

Thus, the candidate score is derived from rubric-level predictions rather than an independent scalar head. The evidence head grounds each judgment in the supplied evidence; auxiliary diagnostics are assembled deterministically rather than generated free-form.

4.2 VERIFIED RELATION CONSTRUCTION

We construct candidate groups so that each task provides both reward-changing and reward-preserving relations. Registered, redistributable Office sources provide task specifications and atomic rubrics. Deterministic edits produce six candidates: a correct candidate, a minimal failure, a partial completion, a compound failure, and reward-preserving equivalents of the correct and minimal-failure candidates. These yield pointwise rubric labels, four reward-changing relations, and two reward-preserving relations. All learned methods use the same candidate set.

A task-aware verifier and an independent package-level oracle separately compute each candidate's rubric reward vector. Groups are retained only when the two evaluations agree. The verified vectors determine relation type, relation order, and the affected and unaffected rubric sets. A reward-changing relation identifies rubric outcomes that differ, whereas a reward-preserving relation requires the complete rubric reward vector to remain unchanged.

A frozen rendering pipeline additionally identifies strict artifact-alias subset by checking visual equality. Relational objectives aren't restricted to exact-visual subset. Sources are split by family and duplicate component before candidate generation, and external benchmarks remain test-only.

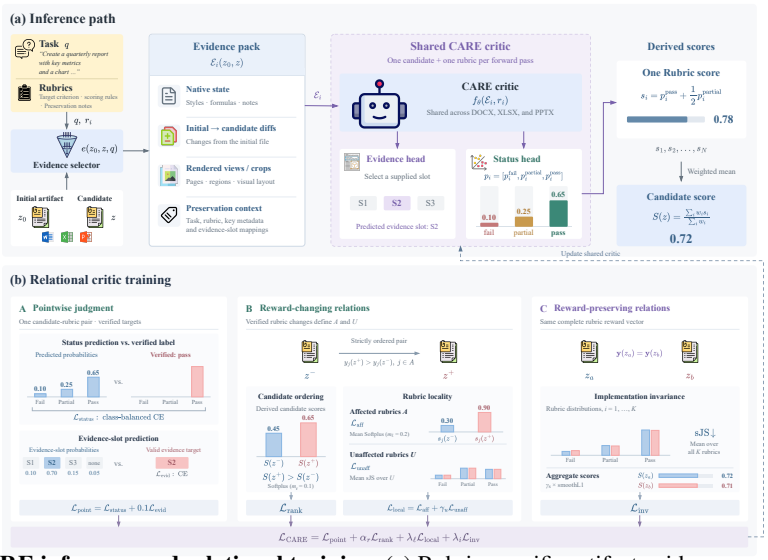


Figure 2: CARE inference and relational training. (a) Rubric-specific artifact evidence supports status and evidence-slot predictions, with rubric scores aggregated into a candidate score. (b) Verified targets supervise pointwise judgment; reward-changing pairs supervise ranking and locality, while reward-preserving pairs supervise invariance. Relation metadata is used only for training.

4.3 Relational Critic Training

The four objectives supervise complementary properties: pointwise judgment, candidate ordering, localized responses to reward-changing edits, and stability under reward-preserving edits. For each candidate-rubric pair, the pointwise objective is

$$\mathcal{L}_{\text{point}} = \mathcal{L}_{\text{status}} + \beta_{\text{evid}}\mathcal{L}_{\text{evid}}, \tag{4}$$

where $\mathcal{L}_{\text{status}}$ is class-balanced cross-entropy and $\mathcal{L}_{\text{evid}}$ is cross-entropy over evidence slots when a valid evidence target is available. For a strictly ordered pair (z^+, z^-), we use

$$\mathcal{L}_{\text{rank}} = \text{softplus}\left[-\left(S(z^+) - S(z^-) - m_r\right)\right]. \tag{5}$$

Reward-changing relations provide rubric-level supervision beyond aggregate ordering. Let A and U denote affected and unaffected rubric sets. Locality relation is oriented as (z^+, z^-) such that

$$y_j(z^+) > y_j(z^-), \qquad \forall j \in A.$$

The locality objective encourages the corresponding predicted scores to increase while keeping unaffected predictions stable:

$$\mathcal{L}_{\text{aff}} = \frac{1}{|A|}\sum_{j\in A}\text{softplus}\left[-\left(s_j(z^+) - s_j(z^-) - m_\ell\right)\right], \tag{6}$$

$$\mathcal{L}_{\text{unaff}} = \frac{1}{|U|}\sum_{i\in U}\text{sJS}\left(\boldsymbol{p}_i^+, \boldsymbol{p}_i^-\right), \tag{7}$$

$$\mathcal{L}_{\text{local}} = \mathcal{L}_{\text{aff}} + \gamma_u\mathcal{L}_{\text{unaff}}. \tag{8}$$

When $U = \varnothing$, we set $\mathcal{L}_{\text{unaff}} = 0$. Here

$$\text{sJS}(\boldsymbol{p}, \boldsymbol{q}) = \tfrac{1}{2}\,\text{KL}(\boldsymbol{p}\|\boldsymbol{m}) + \tfrac{1}{2}\,\text{KL}(\boldsymbol{q}\|\boldsymbol{m}), \qquad \boldsymbol{m} = \tfrac{1}{2}(\boldsymbol{p}+\boldsymbol{q}). \tag{9}$$

Affected and unaffected masks are supervision data used by sampler and loss and are never serialized into model input. For a reward-preserving implementation pair (z_a, z_b) satisfying $\boldsymbol{y}(z_a) = \boldsymbol{y}(z_b)$, we penalize both rubric-distribution drift and aggregate-score drift:

$$\mathcal{L}_{\text{inv}} = \frac{1}{K}\sum_{i=1}^{K}\text{sJS}\left(\boldsymbol{p}_i^a, \boldsymbol{p}_i^b\right) + \gamma_s\,\text{smoothL1}\left(S(z_a), S(z_b)\right). \tag{10}$$

Both passing and failing reward-preserving pairs are included so that an equivalence transformation can't itself serve as success marker. We use **causally grounded** in restricted sense relation supervision arises from controlled artifact interventions whose rubric effects are independently verified.

The full objective is

$$\mathcal{L}_{\text{CARE}} = \underbrace{\mathcal{L}_{\text{point}} + \alpha_r \mathcal{L}_{\text{rank}}}_{\text{judgment and ranking}} + \underbrace{\lambda_\ell \mathcal{L}_{\text{local}} + \lambda_i \mathcal{L}_{\text{inv}}}_{\text{selective scoring responses}} . \tag{11}$$

Learned variants progressively enable these objectives: Artifact-SFT uses $\mathcal{L}_{\text{point}}$; Artifact Pairwise-RM additionally uses $\mathcal{L}_{\text{rank}}$; CARE without invariance additionally uses $\mathcal{L}_{\text{local}}$; and full CARE uses Eq. 11. Relation-aware minibatching and deferred backpropagation are described in Appendix B.

5 Experimental Setup

Datasets and metrics. Dataset contains source, task-mutation, and synthetic executor-style test splits with 9,018, 2,214, and 8,532 rubric records. We use status accuracy and macro-F1. Exact-visual alias accuracy measures ordering on reward-changing artifact aliases and assigns half credit to exact ties. Affected margins measure directional responses on changed rubrics; unaffected drift measures prediction changes on unchanged rubrics within reward-changing relations; and preserving drift measures candidate-score changes across reward-preserving relations. Scalar drift is computed from expected fail/partial/pass rewards, while JS divergence measures changes in the full status distribution. Alias metrics use the exact-visual subset, whereas drift metrics use all eligible relations in each split.

Baselines. Artifact-SFT, Artifact Pairwise-RM, CARE without invariance, and full CARE use Qwen3-VL-8B-Instruct (Bai et al., 2025a). The four differ only in the enabled objectives. Artifact-SFT uses $\mathcal{L}_{\text{point}}$, Pairwise-RM additionally uses $\mathcal{L}_{\text{rank}}$, CARE without invariance adds $\mathcal{L}_{\text{local}}$, and full CARE additionally uses $\mathcal{L}_{\text{inv}}$.

Observability and analysis controls. To isolate evidence access from learned response behavior, exploratory open-model controls compare render-only and render-plus-structure inputs using Qwen3-VL-8B, Qwen3-VL-32B, and Qwen2.5-VL-7B (Bai et al., 2025b) without CARE fine-tuning. ZeroGUI-style (Yang et al., 2025) visual-outcome and UPE-style (Lin et al., 2025) three-prompt judges are adapted for final-artifact evaluation. Five input interventions are evaluated on the same frozen full CARE checkpoints to test dependence on matched native and visual evidence.

Training details. Training uses NF4 QLoRA (Dettmers et al., 2023) with LoRA adapters (Hu et al., 2022), rank 8, learning rate 5×10^{-5}, 3 seeds, and at most 3 images. All enabled outer relation objectives have unit weight; remaining margins and loss coefficients are reported in Appendix B.

OfficeVal. We evaluate fixed-pool Best-of-4 selection on 84 OfficeVal tasks (Zhou et al., 2026), using four fixed Qwen3-Coder-30B-A3B-Instruct attempts per task. All selectors operate on the same frozen pool of 336 candidate slots, including 56 no-delivery failures, and use the official task scoring. Training settings and metric definitions are given in Appendix B.

6 Results and Analysis

6.1 Native evidence improves observability of hidden task states

We isolate **observability** question using exploratory open-model controls: does native Office structure provide task-relevant information that render observations omit? Across all models and test splits, adding native structure to rendered evidence increases macro-F1 as shown in Figure 3. The improvement isn't specific to a single model or split.

Exact-visual artifact aliases isolate observation gap more directly. On 1,141 source-split aliases, all render-only judges assign identical scores to both endpoints and tie every pair. Because alias accuracy assigns half credit to exact ties, this yields an accuracy of 0.5 without resolving any hidden-state difference. With native structure, Qwen3-VL-8B, Qwen3-VL-32B, and Qwen2.5-VL-7B correctly

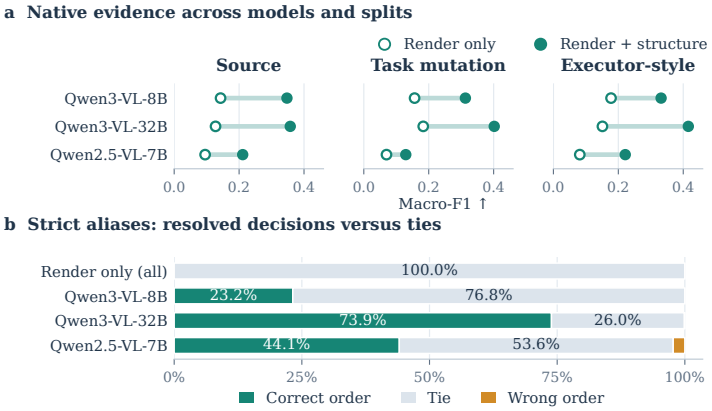


Figure 3: Native evidence improves observability without CARE fine-tuning. Top: macro-F1 for render-only and render-plus-structure inputs across three models and three test splits. Bottom: ordering outcomes on 1,141 exact-visual reward-changing aliases from the source split. Render-only judges tie every alias; adding structure resolves a subset of these pairs into correct or incorrect orderings. Alias accuracy assigns half credit to exact ties, so a render-only value of 0.5 reflects complete ambiguity rather than successful discrimination.

order 23.2%, 73.9%, and 44.1% of pairs, respectively, while their tie rates fall to 76.8%, 26.0%, and 53.6%. Thus, native state exposes reward-relevant distinctions that are unavailable from matched rendered observations alone.

However, access to this signal doesn't guarantee that a critic applies it correctly to corresponding rubric: substantial ties remain for some models even after structure is provided. This separates evidence availability from response learning. We next hold artifact evidence fixed and test how different training objectives shape rubric-level responses to verified artifact changes. Complete classification and alias results are in Tables 3 and 4 in Appendix. The learned and baseline prompt contracts are provided in Appendix F; exploratory open-model calls doesn't retain standalone prompts.

6.2 Locality constrains reward-changing responses to affected rubrics

Table 1: F1 is status macro-F1; Alias is accuracy on exact-visual changing pairs with half credit for ties.

	Source		Task mutation		Executor-style	
Method	F1↑	Alias↑	F1↑	Alias↑	F1↑	Alias↑
Artifact-SFT	$0.858_{0.000}$	$0.974_{0.003}$	$0.679_{0.007}$	$0.900_{0.069}$	$0.812_{0.004}$	$0.916_{0.003}$
Pairwise-RM	$0.836_{0.008}$	$0.969_{0.001}$	$0.661_{0.025}$	$0.972_{0.009}$	$0.828_{0.008}$	$0.926_{0.005}$
CARE w/o inv.	$0.853_{0.013}$	$0.973_{0.004}$	$0.671_{0.015}$	$0.962_{0.019}$	$0.814_{0.009}$	$0.917_{0.006}$
CARE (full)	$0.801_{0.009}$	$0.974_{0.004}$	$0.668_{0.043}$	$0.918_{0.027}$	$0.764_{0.014}$	$0.920_{0.005}$

We hold artifact evidence fixed and ask whether supervision determines **where** score changes occur across related candidates. Table 1 first establishes basic judgment quality under this matched setting. Adding locality to Artifact Pairwise-RM increases mean macro-F1 from 0.836 to 0.853 on source and from 0.661 to 0.671 on task mutation, while changing from 0.828 to 0.814 on synthetic executor-style data. Alias accuracy changes from 0.969 to 0.973 on source, from 0.972 to 0.962 on task mutation, and from 0.926 to 0.917 on executor-style data. These classification and ranking measures remain distinct from the rubric-level drift measured below.

Aggregate ranking alone does not localize the reward response. Relative to Artifact-SFT, adding the candidate-ranking objective increases unaffected absolute drift from 0.027 to 0.226 on source, from 0.188 to 0.321 on task mutation, and from 0.040 to 0.135 on executor-style data. A candidate-level ordering can therefore be learned while predictions for unchanged rubrics move substantially. Adding the locality objective reverses much of this increase as shown in Figure 4 a. Artifact-SFT still has lower unaffected scalar drift than the locality model on source and executor-style data, so the relevant comparison here is the incremental effect of locality after candidate-ranking supervision.

The lower unaffected drift does not result from suppressing all responses. For CARE without invariance, affected-rubric margins remain positive at 0.703, 0.553, and 0.536 on source, task mutation, and executor-style data. On source, locality increases affected margin from 0.688 to 0.703

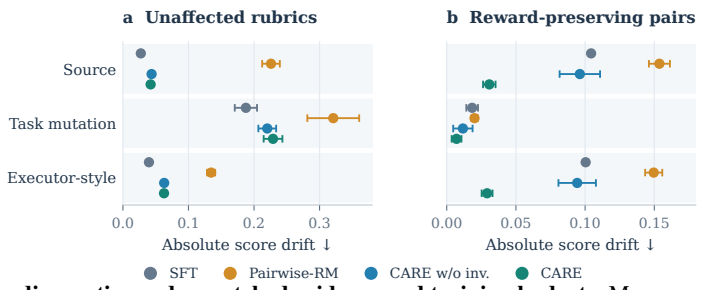


Figure 4: Response diagnostics under matched evidence and training budgets. Means and sample SDs over 3 seeds are shown for (a) unaffected rubric-score drift on reward-changing relations and (b) candidate-score drift on reward-preserving relations. All eligible relations in each split enter these diagnostics. Figure (a) isolates the effect of locality and figure (b) isolates the effect of invariance.

while reducing unaffected JS from 0.143 to 0.055. On task mutation and executor-style data, affected margins are smaller than those of Pairwise-RM but remain positive. Despite mixed changes in macro-F1, these results indicate that locality primarily changes **scope** of reward response: it reduces collateral changes in unaffected rubrics while retaining verified response direction on affected rubrics. The Full CARE prompt used by all learned variants is provided in Appendix F.

6.3 Invariance stabilizes reward-preserving implementations

Locality constrains how predictions respond when rubric rewards change. We test complementary question: when an implementation changes but the complete rubric reward vector remains unchanged, does critic preserve its predictions? We isolate this effect by comparing CARE without invariance with full CARE, which differ only by $\mathcal{L}_{\text{inv}}$ in the matched learning comparison.

Adding invariance reduces candidate-score drift on reward-preserving pairs shown in Figure 4 b. Mean preserving drift decreases from 0.096 to 0.031 on source, from 0.012 to 0.007 on task mutation, and from 0.094 to 0.029 on synthetic executor-style data. The largest absolute reductions occur on source and executor-style data, where implementation-equivalent candidates otherwise receive noticeably different aggregate scores. Full CARE has lower preserving drift than Artifact-SFT, indicating invariance objective does more than recover stability lost through candidate-ranking supervision.

The stability gain comes with a classification cost in the recorded matched comparison. Relative to CARE without invariance, full CARE reduces mean macro-F1 from 0.853 to 0.801 on source, from 0.671 to 0.668 on task mutation, and from 0.814 to 0.764 on executor-style data. Alias accuracy changes from 0.973 to 0.974, from 0.962 to 0.918, and from 0.917 to 0.920, respectively. Affected-rubric margins remain positive at 0.701, 0.540, and 0.550. Thus, invariance improves stability on reward-preserving implementations while retaining positive affected-rubric margins. Mean macro-F1 is lower on all datasets, with the largest decreases on source and executor-style data; alias accuracy is lower on task mutation. Full CARE prompt is unchanged for experiment; it is provided in Appendix F.

6.4 CARE depends on matched native evidence and complementary renders

The open-model controls in Section 6.1 show that native Office state can expose task information missing from rendered observations. We test whether a trained CARE critic actually depends on candidate-matched native evidence. We intervene only on the inputs to the same full-CARE model, using five views: the final screen, full renders, structure only, matched render plus structure, and render plus structure from a different candidate. All five views omit crops and evidence-location mappings. They therefore use different inputs from Table 1; compare their absolute macro-F1 values within this input-view experiment.

Native structure carries most of the predictive signal in these tasks. Structure-only inputs achieve macro-F1 values of 0.770, 0.539, and 0.738 on source, task mutation, and synthetic executor-style data, respectively, exceeding both render-only views in every split. Rendered evidence nevertheless remains complementary: matched render+structure increases these values to 0.809, 0.669, and 0.768. The largest increment occurs on task mutation, where the mean rises from 0.539 with structure alone to 0.669 with matched multimodal evidence.

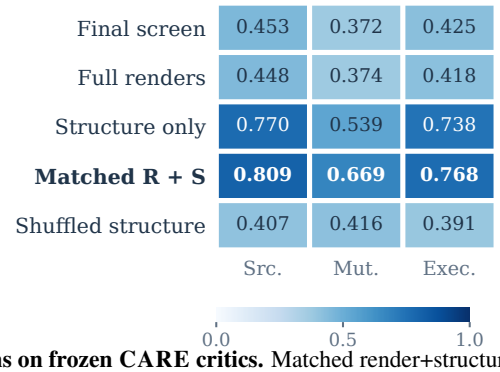


Figure 5: Input interventions on frozen CARE critics. Matched render+structure performs best; substituting another candidate's native structure reduces macro-F1. R and S denote renders and structure.

The native evidence must also correspond to the candidate being judged. Replacing the matched structure with structure from another candidate reduces macro-F1 from 0.809 to 0.407 on source, from 0.669 to 0.416 on task mutation, and from 0.768 to 0.391 on executor-style data. Together, the controls indicate that CARE relies on candidate-specific native state while also benefiting from complementary rendered information. Full CARE prompt is shown in Appendix F.

6.5 Fixed-pool OfficeVal tests candidate selection

Table 2: OfficeVal Best-of-4 on 84 tasks. Scores are official normalized task scores.

Method	Score↑	Gain over random
Random	0.05663	–
ZeroGUI-style	0.05790	+0.00127
UPE-style	0.06090	+0.00427
Artifact-SFT	0.06406	+0.00743
Artifact Pairwise-RM	0.06285	+0.00621
CARE w/o invariance	0.06705	+0.01042
CARE (full)	**0.06999**	**+0.01336**
Verifier oracle	0.09225	+0.03562

We evaluate whether critic scores support selection among externally generated deliverables using the same four fixed attempts for each of 84 OfficeVal tasks. As shown in Table 2, full CARE achieves the highest mean score among the learned methods at 0.06999, followed by CARE without invariance at 0.06705, Artifact-SFT at 0.06406, and Artifact Pairwise-RM at 0.06285. The two CARE variants exceed random selection by 0.01336 and 0.01042, while full CARE exceeds Artifact-SFT by approximately 0.00593 based on the displayed means. The verifier oracle scores 0.09225, leaving further room for improved selection within the existing pool. The learned selector uses the full-CARE prompt, while the ZeroGUI-style and UPE-style selectors use the baseline prompts in Appendix F. Overall, the results support the utility of CARE for fixed-pool artifact selection.

7 Conclusion

We introduce CARE, a rubric-conditioned Office artifact critic that combines native and rendered evidence with programmatically verified relation supervision, without requiring a golden artifact at inference. Under matched evidence and training budgets, locality reduces unaffected-rubric drift relative to Artifact Pairwise-RM, while invariance stabilizes scores across reward-preserving implementations. These gains coexist with positive mean affected-rubric margins, distinguishing selective stability from uniformly suppressing score changes. Input interventions further show the importance of candidate-matched native evidence and complementary rendered information. On fixed-pool OfficeVal Best-of-4 selection, full CARE achieves the highest mean official score among the learned selectors. Together, these findings distinguish access to task-relevant state from learning how judgments should respond to its changes, supporting joint evaluation of rubric-level correctness, response stability, and downstream selection rather than treating these properties as interchangeable.

AI USE STATEMENT

We use generative AI tools exclusively for language editing, grammatical checking, and manuscript proofreading. It isn't involved in data analysis, methodological implementation, or conceptual reasoning. It doesn't supply or replace the human eligibility labels. Frozen labels, predictions, thresholds, and confirmatory decision are strictly preserved. The authors are fully responsible for verifying the analyses, interpretations, citations, and final text; AI assistance doesn't substitute for their scientific judgment.

REFERENCES

David Anugraha, Zilu Tang, Lester James Validad Miranda, Hanyang Zhao, and Shou-Yi Hung. R3: Robust Rubric-Agnostic Reward Models. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, and Xionghui Chen. Qwen3-VL Technical Report, 2025a.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, and Wenbin Ge. Qwen2.5-VL Technical Report, 2025b.

Léo Boisvert, Megh Thakkar, Maxime Gasse, Massimo Caccia, and Thibault Le Sellier De Chezelles. WorkArena++: Towards Compositional Planning and Reasoning-based Common Knowledge Work Tasks. In *Advances in Neural Information Processing Systems(NIPS)*, volume 37, pp. 5996–6051, 2024.

Paul F. Christiano, Jan Leike, Tom B. Brown, Miljan Martic, and Shane Legg. Deep Reinforcement Learning from Human Preferences. In *Advances in Neural Information Processing Systems(NIPS)*, 2017.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient Finetuning of Quantized LLMs. In *Advances in Neural Information Processing Systems(NIPS)*, volume 36, 2023.

Alexandre Drouin, Maxime Gasse, Massimo Caccia, Issam H. Laradji, and Manuel Del Verme. WorkArena: How Capable are Web Agents at Solving Common Knowledge Work Tasks? In *Proceedings of the 41st International Conference on Machine Learning(ICML)*, volume 235, pp. 11642–11662, 2024.

Ecma International. Office Open XML File Formats—ECMA-376, 2016.

Matt Gardner, Yoav Artzi, Victoria Basmov, Jonathan Berant, and Ben Bogin. Evaluating Models' Local Decision Boundaries via Contrast Sets. In *Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2020(EMNLP)*, pp. 1307–1323, 2020.

Anwen Hu, Haiyang Xu, Jiabo Ye, Ming Yan, and Liang Zhang. mPLUG-DocOwl 1.5: Unified Structure Learning for OCR-free Document Understanding. In *Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2024(EMNLP)*, pp. 3096–3120, 2024.

Anwen Hu, Haiyang Xu, Liang Zhang, Jiabo Ye, and Ming Yan. mPLUG-DocOwl2: High-resolution Compressing for OCR-free Multi-page Document Understanding. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)(ACL)*, pp. 5817–5834, 2025.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, and Yuanzhi Li. LoRA: Low-Rank Adaptation of Large Language Models. In *Proceedings of the 10th International Conference on Learning Representations(ICLR)*, 2022.

Yupan Huang, Tengchao Lv, Lei Cui, Yutong Lu, and Furu Wei. LayoutLMv3: Pre-training for Document AI with Unified Text and Image Masking. In *Proceedings of the 30th ACM International Conference on Multimedia(ACM MM)*, 2022.

Yuxin Jiang, Yufei Wang, Xingshan Zeng, Wanjun Zhong, and Liangyou Li. FollowBench: A Multi-level Fine-grained Constraints Following Benchmark for Large Language Models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)(ACL)*, pp. 4667–4688, 2024.

Divyansh Kaushik, Eduard Hovy, and Zachary C. Lipton. Learning the Difference that Makes a Difference with Counterfactually-Augmented Data. In *Proceedings of the 8th International Conference on Learning Representations(ICLR)*, 2020.

Seungone Kim, Jamin Shin, Yejin Cho, Joel Jang, and Shayne Longpre. Prometheus: Inducing Fine-Grained Evaluation Capability in Language Models. In *Proceedings of the 12th International Conference on Learning Representations(ICLR)*, 2024a.

Seungone Kim, Juyoung Suk, Shayne Longpre, Bill Yuchen Lin, and Jamin Shin. Prometheus 2: An Open Source Language Model Specialized in Evaluating Other Language Models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing(EMNLP)*, pp. 4334–4353, 2024b.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, and Bowen Baker. Let's Verify Step by Step. In *Proceedings of the 12th International Conference on Learning Representations(ICLR)*, 2024.

Haojia Lin, Xiaoyu Tan, Yulei Qin, Zihan Xu, and Yuchen Shi. CUARewardBench: A Benchmark for Evaluating Reward Models on Computer-using Agent, 2025.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, and Xuanyu Lei. AgentBench: Evaluating LLMs as Agents. In *Proceedings of the 12th International Conference on Learning Representations(ICLR)*, 2024.

Yantao Liu, Zijun Yao, Rui Min, Yixin Cao, and Lei Hou. RM-Bench: Benchmarking Reward Models of Language Models with Subtlety and Style. In *Proceedings of the 13th International Conference on Learning Representations(ICLR)*, 2025.

Zeyao Ma, Bohan Zhang, Jing Zhang, Jifan Yu, and Xiaokang Zhang. SpreadsheetBench: Towards Challenging Real World Spreadsheet Manipulation. In *Advances in Neural Information Processing Systems(NIPS)*, volume 37, 2024.

Bingchen Miao, Yang Wu, Minghe Gao, Qifan Yu, and Wendong Bu. Boosting Virtual Agent Learning and Reasoning: A Step-Wise, Multi-Dimensional, and Generalist Reward Model with Benchmark. In *Proceedings of the 42nd International Conference on Machine Learning(ICML)*, volume 267, pp. 44054–44075, 2025.

Michael Ofengenden, Yunze Man, Ziqi Pang, Liang-Yan Gui, and Yu-Xiong Wang. PPTArena: A Benchmark for PowerPoint Editing, 2025.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, and Carroll Wainwright. Training Language Models to Follow Instructions with Human Feedback. In *Advances in Neural Information Processing Systems(NIPS)*, 2022.

Yiwei Qin, Kaiqiang Song, Yebowen Hu, Wenlin Yao, and Sangwoo Cho. InFoBench: Evaluating Instruction Following Ability in Large Language Models. In *Proceedings of the Findings of the Association for Computational Linguistics: ACL 2024(ACL)*, pp. 13025–13048, 2024.

Chris Rawles, Sarah Clinckemaillie, Yifan Chang, Jonathan Waltz, and Gabrielle Lau. AndroidWorld: A Dynamic Benchmarking Environment for Autonomous Agents. In *Proceedings of the 13th International Conference on Learning Representations(ICLR)*, 2025.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. Beyond Accuracy: Behavioral Testing of NLP Models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics(ACL)*, pp. 4902–4912, 2020.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel M. Ziegler, and Ryan Lowe. Learning to Summarize from Human Feedback. In *Advances in Neural Information Processing Systems(NIPS)*, 2020.

Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, and Noah Siegel. Solving math word problems with process- and outcome-based feedback, 2022.

Haoxiang Wang, Wei Xiong, Tengyang Xie, Han Zhao, and Tong Zhang. Interpretable Preferences via Multi-Objective Reward Modeling and Mixture-of-Experts. In *Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2024(EMNLP)*, pp. 10582–10592, 2024.

Bosi Wen, Pei Ke, Xiaotao Gu, Lindong Wu, and Hao Huang. Benchmarking Complex Instruction-Following with Multiple Constraints Composition. In *Advances in Neural Information Processing Systems(NIPS)*, volume 37, pp. 137610–137645, 2024.

Zhaofeng Wu, Michihiro Yasunaga, Andrew Cohen, Yoon Kim, and Asli Celikyilmaz. reWordBench: Benchmarking and Improving the Robustness of Reward Models with Transformed Inputs. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing(EMNLP)*, pp. 3383–3409, 2025.

Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. Unsupervised Data Augmentation for Consistency Training. In *Advances in Neural Information Processing Systems(NIPS)*, volume 33, pp. 6256–6268, 2020.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, and Siheng Zhao. OSWorld: Benchmarking Multimodal Agents for Open-Ended Tasks in Real Computer Environments. In *Advances in Neural Information Processing Systems(NIPS)*, 2024.

Chenyu Yang, Shiqian Su, Shi Liu, Xuan Dong, and Yue Yu. ZeroGUI: Automating Online GUI Learning at Zero Human Cost, 2025.

Hao Zheng, Xinyan Guan, Hao Kong, Wenkai Zhang, and Jia Zheng. PPTAgent: Generating and Evaluating Presentations Beyond Text-to-Slides. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing(EMNLP)*, pp. 14402–14418, 2025.

Jingbo Zhou, Yusai Zhao, Qi Bao, Jingjia Cao, and Zhenghai Chen. OmegaUse-OfficeVal: Benchmarking LLM Agents on Long-Horizon Office-Suite Tasks with Economic Grounding, 2026.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, and Robert Lo. WebArena: A Realistic Web Environment for Building Autonomous Agents. In *Proceedings of the 12th International Conference on Learning Representations(ICLR)*, 2024.

Appendix Contents

A The Use of Large Language Models 14

B Training settings and metric definitions 14

C Open-model results with and without native structure 14

D Additional results 14

E Exploratory OfficeVal open-model results 16

F Prompts 16

A THE USE OF LARGE LANGUAGE MODELS

We used generative AI tools to improve wording and grammar and to help with LaTeX formatting and document preparation. The authors are responsible for checking the final text, references, figures, tables, and files. The reported results come from the existing experiments; AI-generated text is not used as human annotation or as independent experimental evidence.

B TRAINING SETTINGS AND METRIC DEFINITIONS

All four trained methods use the checkpoint saved after 1,781 steps. Each method sees the same candidate order and images. The full dataset contains 5,784 groups and 104,112 rubric inputs across six training, development, and test datasets.

Setting	Value
Backbone	Qwen3-VL-8B-Instruct, NF4 weights, BF16 compute
LoRA	Rank 8, alpha 16, dropout 0.05; attention and MLP projections; visual encoder excluded
Input	8,192 tokens; at most 3 images; 50,176–200,704 pixels per image
Optimization	Learning rate 5×10^{-5}; linear decay; 32 warmup steps; weight decay 0; gradient norm clip 1
Execution	Evidence microbatch 2; 1,781 optimizer steps; two-pass back-propagation for relation losses
Pointwise weights	Fail/partial/pass: 1.5, 6.0, $6/13$; location coefficient $\beta_{\text{loc}} = 0.1$
Relation coefficients	Enabled losses have weight 1; $\gamma_u = \gamma_s = 1$; $m_r = 0.1$, $m_\ell = 0.2$

Metrics Definetion Unaffected drift is $|s_i^a - s_i^b|$ for rubric items whose correct status stays the same when a candidate is edited. Preserving drift is $|S(z_a) - S(z_b)|$ for candidate pairs whose full rubric reward vector stays the same. Unaffected JS measures changes in the full pass/partial/fail distribution, using the same divergence as the training loss. Alias ranking uses pairs with identical rendered evidence.

C OPEN-MODEL RESULTS WITH AND WITHOUT NATIVE STRUCTURE

Table 3: Open-model classification results. Each cell gives accuracy / macro-F1. Source, mutation, and executor-style splits contain 9,018, 2,214, and 8,532 rubric records. Executor-style examples are synthetic. These exploratory results are point estimates without significance tests.

Model	Evidence	Source	Mutation	Executor-style
Qwen3-VL-8B	Render	0.219 / 0.143	0.154 / 0.156	0.261 / 0.178
Qwen3-VL-8B	Render+structure	0.682 / 0.347	0.679 / 0.313	0.654 / 0.332
Qwen3-VL-32B	Render	0.228 / 0.127	0.285 / 0.183	0.282 / 0.151
Qwen3-VL-32B	Render+structure	0.524 / 0.358	0.612 / 0.402	0.606 / 0.416
Qwen2.5-VL-7B	Render	0.078 / 0.095	0.068 / 0.070	0.075 / 0.081
Qwen2.5-VL-7B	Render+structure	0.343 / 0.211	0.172 / 0.129	0.356 / 0.221

The saved prompts for the trained models and visual baselines are in Appendix F.

D ADDITIONAL RESULTS

The tables below add status accuracy, tie rates for visually identical pairs, score differences on affected rubrics, and changes in predicted status distributions. Relation metrics use all eligible pairs. The input-view comparison omits image crops and mappings from evidence to document locations.

Table 4: Score changes for source-split candidate pairs. Alias Acc. gives half credit to ties; Alias Tie is the tie rate among 1,141 visually identical pairs with different rewards. Affected is the signed score difference on changed rubric items. Unaffected and Preserving are absolute score changes on unchanged rubric items and on candidate pairs with unchanged rewards. These last three measures use all eligible source pairs.

Model	Evidence	Alias Acc.↑	Alias Tie	Affected↑	Unaffected↓	Preserving↓
Qwen3-VL-8B	Render	0.500	1.000	0.002	0.008	0.000
Qwen3-VL-8B	Render+structure	0.616	0.768	0.209	0.103	0.018
Qwen3-VL-32B	Render	0.500	1.000	-0.000	0.005	0.000
Qwen3-VL-32B	Render+structure	0.869	0.260	0.615	0.439	0.045
Qwen2.5-VL-7B	Render	0.500	1.000	-0.000	0.008	0.000
Qwen2.5-VL-7B	Render+structure	0.709	0.536	0.122	0.083	0.050

Table 5: This table gives status accuracy and the tie rate for visually identical pairs. Macro-F1 and alias accuracy are reported in the main text.

Method	Accuracy↑	Alias Tie↓
Source		
Artifact-SFT	0.915 ± 0.000	0.015 ± 0.001
Artifact Pairwise-RM	0.891 ± 0.005	0.015 ± 0.001
CARE without invariance	0.911 ± 0.010	0.014 ± 0.000
CARE (full)	0.871 ± 0.008	0.015 ± 0.001
Task mutation		
Artifact-SFT	0.752 ± 0.008	0.003 ± 0.000
Artifact Pairwise-RM	0.731 ± 0.046	0.003 ± 0.000
CARE without invariance	0.747 ± 0.008	0.003 ± 0.000
CARE (full)	0.744 ± 0.045	0.003 ± 0.000
Synthetic executor-style		
Artifact-SFT	0.858 ± 0.004	0.038 ± 0.005
Artifact Pairwise-RM	0.876 ± 0.004	0.038 ± 0.005
CARE without invariance	0.860 ± 0.005	0.039 ± 0.003
CARE (full)	0.823 ± 0.012	0.035 ± 0.004

Table 6: Affected is the signed score difference on changed rubric items. Unaff. score and Unaff. JS measure changes in the expected score and full status distribution of unchanged items. Preserv. score measures changes in the total score when rewards are unchanged

Method	Affected↑	Unaff. score↓	Unaff. JS↓	Preserv. score↓
Source				
Artifact-SFT	0.660 ± 0.002	0.027 ± 0.001	0.064 ± 0.000	0.104 ± 0.000
Artifact Pairwise-RM	0.688 ± 0.002	0.226 ± 0.013	0.143 ± 0.006	0.154 ± 0.008
CARE without invariance	0.703 ± 0.007	0.044 ± 0.001	0.055 ± 0.003	0.096 ± 0.015
CARE (full)	0.701 ± 0.006	0.042 ± 0.001	0.053 ± 0.001	0.031 ± 0.005
Task mutation				
Artifact-SFT	0.492 ± 0.024	0.188 ± 0.017	0.170 ± 0.029	0.018 ± 0.004
Artifact Pairwise-RM	0.580 ± 0.047	0.321 ± 0.040	0.225 ± 0.043	0.020 ± 0.002
CARE without invariance	0.553 ± 0.044	0.220 ± 0.014	0.178 ± 0.003	0.012 ± 0.007
CARE (full)	0.540 ± 0.037	0.229 ± 0.014	0.172 ± 0.018	0.007 ± 0.004
Synthetic executor-style				
Artifact-SFT	0.491 ± 0.016	0.040 ± 0.001	0.092 ± 0.001	0.100 ± 0.001
Artifact Pairwise-RM	0.605 ± 0.008	0.135 ± 0.006	0.125 ± 0.003	0.150 ± 0.006
CARE without invariance	0.536 ± 0.017	0.063 ± 0.002	0.081 ± 0.002	0.094 ± 0.014
CARE (full)	0.550 ± 0.019	0.063 ± 0.002	0.078 ± 0.002	0.029 ± 0.004

Table 7: Changing the input to the same Full CARE model. Values are macro-F1. All five views omit image crops and mappings to document locations.

Input view	Source	Task mutation	Executor-style
Final screen	0.453 ± 0.011	0.372 ± 0.016	0.425 ± 0.006
Full renders	0.448 ± 0.018	0.374 ± 0.034	0.418 ± 0.018
Structure only	0.770 ± 0.011	0.539 ± 0.066	0.738 ± 0.014
Matched render+structure	0.809 ± 0.012	0.669 ± 0.029	0.768 ± 0.012
Render+shuffled structure	0.407 ± 0.021	0.416 ± 0.006	0.391 ± 0.015

All trained models and input-view comparisons in this section use the Full CARE prompt in Appendix F.

E Exploratory OfficeVal open-model results

These exploratory open-model tests use the same fixed candidates and 84 tasks as the main OfficeVal experiment. The final column repeats evaluation after excluding the same three tasks with verifier errors for every method.

Table 8: Exploratory OfficeVal selection results for open models. Gain is the difference in mean score from uniform random selection on 84 tasks.

Model	Evidence	Gain (84 tasks)	Score (81 tasks)
Qwen3-VL-8B	Render	0.00465	0.06356
Qwen3-VL-8B	Render+structure	0.00712	0.06612
Qwen3-VL-32B	Render	0.00470	0.06361
Qwen3-VL-32B	Render+structure	0.00165	0.06044
Qwen2.5-VL-7B	Render	0.00933	0.06841
Qwen2.5-VL-7B	Render+structure	-0.00029	0.05844

F Prompts

This section lists the prompts used by the trained critic and the two visual judging baselines. At inference time, the fields in braces are filled with the task, rubric, evidence, and document records.

F.1 Full CARE prompt

Full CARE prompt

```
You are judging exactly one rubric for an Office artifact.
Use only the supplied normalized changes and renders.

[TASK]
{instruction}

[RUBRIC]
{rubric_text}

[ARTIFACT SUMMARY]
{compact artifact_summary JSON}

[NORMALIZED CHANGES]
[E00] {event JSON}
[E01] {event JSON}
...

[RENDERS]
[RENDER_00] {uri}
[CROP_00] {uri}
...
```

```
[EVIDENCE LOCATION MAPPING]
{compact mapping JSON}

[RUNTIME IMAGE BUDGET]
{visible_count}/{all_count} frozen images supplied

[GLOBAL PRESERVATION SUMMARY]
{compact JSON}

[TRUNCATION]
{compact JSON}

Judge the rubric using the internal status and location heads.
```

All four trained variants use this prompt. They differ in their training losses.

F.2 B1: ZEROGUI-STYLE VISUAL JUDGE

B1 system prompt

```
You are an expert at judging completion of computer-use tasks from
    final visual observations.
```

B1 user prompt

```
Analyze every supplied final render in order. Decide whether the
    stated rubric is fully satisfied in the delivered final state. Use
     partial only when the visible evidence supports incomplete
    satisfaction, and fail when it supports a violation. Do not infer
    hidden state that is not visually observable.

[TASK INSTRUCTION]
{instruction}

[RUBRIC TO JUDGE]
{rubric_text}

[RENDER EVIDENCE ORDER]
RENDER_00
RENDER_01
...

Return exactly one JSON object and no markdown or extra text:
{"status":"pass|partial|fail","confidence":0.0,"evidence":["RENDER_00
    "]}
```

F.3 B2: UPE-STYLE VISUAL JUDGE

B2 system prompt: upe_completion_v1

```
You are a systematic visual outcome evaluator for computer-use tasks.
```

B2 user prompt: upe_completion_v1

```
First identify what visible condition the rubric requires, then
    inspect all supplied final renders and judge completion. Pass
    requires visible evidence that the complete rubric holds; partial
    requires visible incomplete progress; fail requires visible
    contradiction or absence of the required result.

[TASK INSTRUCTION]
{instruction}

[RUBRIC TO JUDGE]
{rubric_text}

[RENDER EVIDENCE ORDER]
RENDER_00
RENDER_01
...

Return exactly one JSON object and no markdown or extra text:
{"status":"pass|partial|fail","confidence":0.0,"evidence":["RENDER_00
    "]}
```

B2 system prompt: upe_false_accept_v1

```
You are a conservative visual auditor whose priority is avoiding false
     acceptance.
```

B2 user prompt: upe_false_accept_v1

```
Try to falsify the rubric using the supplied final renders. Return
    pass only when no visible requirement is missing or contradicted.
    Return partial for incomplete or ambiguous visible completion and
    fail for a clear visible violation. Do not assume unobserved
    changes succeeded.

[TASK INSTRUCTION]
{instruction}

[RUBRIC TO JUDGE]
{rubric_text}

[RENDER EVIDENCE ORDER]
RENDER_00
RENDER_01
...

Return exactly one JSON object and no markdown or extra text:
{"status":"pass|partial|fail","confidence":0.0,"evidence":["RENDER_00
    "]}
```

B2 system prompt: upe_rubric_check_v1

```
You evaluate one task rubric using an explicit visual checklist.
```

B2 user prompt: upe_rubric_check_v1

```
Convert the rubric into its visible checklist items, examine the
    complete ordered render set, and aggregate the checklist: pass if
    every visible item is satisfied, partial if only some are
    satisfied, and fail if the central visible requirement is violated
     or absent.

[TASK INSTRUCTION]
{instruction}

[RUBRIC TO JUDGE]
{rubric_text}

[RENDER EVIDENCE ORDER]
RENDER_00
RENDER_01
...

Return exactly one JSON object and no markdown or extra text:
{"status":"pass|partial|fail","confidence":0.0,"evidence":["RENDER_00
    "]}
```