

SciFigBench: A Benchmark for Scientific Method Figure Understanding

Lijia Wu^{1,2}, Zheng Li^{3,*}, Yuxuan Liu^{1,2},
Jingwen Zhang^{1,2}, Penghai Zhao^{1,2}, Xiang Li^{2,1,*}

¹VCIP, CS & AAIS, Nankai University

²NKIARI, Shenzhen Futian ³Li Auto.

lijiawu@mail.nankai.edu.cn

zhengli97@foxmail.com xiang.li.implus@nankai.edu.cn

Abstract

Scientific method figures often provide researchers with a clear visual entry point into a paper’s methodology and core contributions, yet their dense layouts and complex information flows are difficult for Vision Language Models (VLMs) to interpret. Existing benchmarks predominantly focus on coarse-grained tasks, such as context-based retrieval or holistic comprehension, overlooking the fine-grained content and inter-module relations within complex methodological diagrams. To address these overlooked aspects, we introduce **Scientific Figure Benchmark (SciFigBench)**, a benchmark for evaluating core capabilities required for scientific method figure understanding. Specifically, it targets three capabilities: dense content perception, visual relation comprehension, and information-flow tracing. For this purpose, we curate 4,352 scientific method figures from real-world research papers and construct 40,670 QA pairs to operationalize these capabilities as measurable question-answering tasks. All annotations are generated by dedicated pipelines and then manually verified. We further propose **SciFigCue**, a training-free visual prompting module that externalizes arrow-induced dependencies as visual relational evidence for method-figure reasoning. Experiments across open-source and proprietary VLMs show that even strong models remain substantially limited in visual relation comprehension, while SciFigCue improves performance across several tasks without parameter updates. The code and related resources are available at <https://github.com/LeNcyer/SciFigBench>.

1 Introduction

As AI systems become increasingly involved in scientific research, recent work has explored their potential across research workflows (Garikaparathi et al., 2025; Zhang et al., 2024; Zhao et al., 2026,

*Corresponding authors.

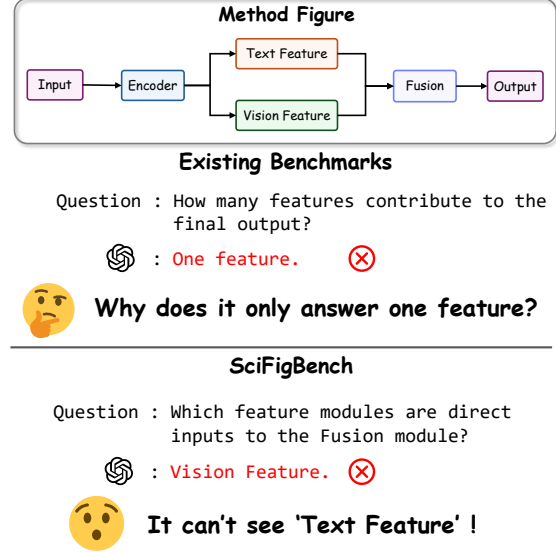


Figure 1: **Why diagnostic method-figure QA is needed.** A coarse-grained question may expose an incorrect answer, but cannot distinguish whether the failure comes from missing a component label or misreading a structural connection. SciFigBench instead asks fine-grained diagnostic questions to make these failure sources observable.

2025a; Wang et al., 2024). To be useful in these settings, such systems must understand the methods described in papers, rather than only retrieve factual information. This requires models to parse how papers organize procedural steps and communicate technical ideas. In research papers, method figures are among the most compact forms of method communication, organizing model architectures, processing pipelines, and technical ideas into a single diagram. Their semantics are not defined by individual visual objects, but by the interaction between text labels, visual modules, and arrow connections. Thus, method figure understanding requires recovering both local content and directed relations, rather than merely recognizing elements in the image.

This demand for fine-grained diagram under-

standing has motivated recent benchmarks on scientific diagrams and method figures in research papers (Zhao et al., 2025b; Pan et al., 2024; Shen et al., 2024). Existing benchmarks cover several evaluation formats, including information-seeking QA over schematic diagrams, holistic figure-level comprehension, and structural reasoning over flowchart-style synthetic diagrams. These works provide important testbeds and show that current VLMs still have gaps in scientific figure understanding.

However, final task scores indicate only whether a model answers correctly, offering little insight into the specific sources of failure in method figure understanding. The same wrong answer may arise from different causes: a model may miss local text, misread an arrow connection, or fail to follow a multi-step information flow. These failures correspond to different visual reasoning abilities, yet existing evaluations usually collapse them into a single aggregate score.

To diagnose such failures, we introduce **SciFigBench**, a **Scientific Figure Benchmark** for evaluating VLMs’ ability to understand real scientific method figures. Specifically, SciFigBench is organized around three complementary evaluation perspectives: (1) **dense content perception**, assessing whether models can capture textual cues distributed across method components; (2) **visual relation comprehension**, measuring their ability to identify module connections, local connectivity patterns, and directional interactions; and (3) **information-flow tracing**, testing whether models can integrate textual cues with structural relations to trace input–output flows across method modules. By separating these perspectives, SciFigBench links model failures to distinct visual reasoning abilities.

Building such a diagnostic benchmark requires structured annotations of figure elements and their directed relations, which are rarely available from real method figures. Therefore, we introduce an automated annotation pipeline that generates initial annotations for figure text, visual components, and their connections. All generated candidate annotations are then manually verified to ensure the quality of the final evaluation data. This pipeline enables the construction of SciFigBench, comprising 4,352 scientific method figures curated from 2024–2025 AI conference papers and 40,670 QA pairs across the three diagnostic dimensions. We use AI method figures as an initial testbed because

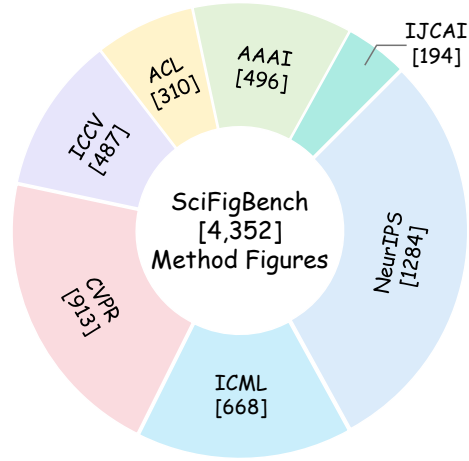


Figure 2: **Source distribution of SciFigBench method figures.** SciFigBench contains 4,352 method figures from papers published at seven AI conferences in 2024–2025.

text-dense, free-form layouts and directed information flows closely align with the figure-internal capabilities targeted by SciFigBench. Figure 2 summarizes the distribution of these method figures across seven venues.

Experiments on SciFigBench reveal that strong proprietary and open-source VLMs still struggle with fine-grained relation understanding and information-flow tracing in scientific method figures. These failures point to arrow-based connection cues as a key difficulty: a model must detect arrows, infer their directions, and compose the induced connections to follow relations across method components.

This diagnosis points to a simple input-level enhancement through visual prompting (Shtedritski et al., 2023): making the missing connection cues explicit. Therefore, we introduce **SciFigCue**, a training-free visual prompting module for scientific method figures. Specifically, SciFigCue renders arrow-based connection cues in two forms. It overlays arrow cues on the original figure to preserve visual context, and provides an auxiliary arrow-segmentation view to make arrow locations and head–tail directions explicit. As a training-free input preprocessing module, SciFigCue can be seamlessly integrated into existing VLMs without additional training cost or model modification. Extensive experiments show that this simple plugin improves multiple connection-dependent tasks, suggesting that explicit connection cues strengthen structured reasoning over method figures.

Our contributions are as follows:

- We introduce **SciFigBench**, a diagnostic benchmark for evaluating VLMs’ figure-internal understanding of real scientific method figures.
- We construct a human-verified dataset with 4,352 real scientific method figures and 40,670 QA pairs.
- We propose **SciFigCue**, a training-free visual prompting module that makes arrow-based connection cues explicit for VLM reasoning.
- Experiments show that current VLMs struggle to compose arrow-induced connections, while SciFigCue’s explicit connection cues help alleviate this bottleneck.

2 Related Work

2.1 Structured Visual Understanding Benchmarks

Structured visual understanding benchmarks have expanded from charts (Masry et al., 2022; Wei et al., 2025) to flowcharts and scientific figures. FlowchartQA (Tannert et al., 2023), FlowVQA (Singh et al., 2024), FlowLearn (Pan et al., 2024), and FlowExplainBench (Suri et al., 2025) focus on reasoning over procedural structures, control flows, and graph-like relations. FlowLearn is closest to our setting, but its fine-grained topology analysis is mainly built on simulated flowcharts, while its real scientific-figure subset is evaluated primarily through OCR, true/false QA, and caption-based description. MISS-QA (Zhao et al., 2025b) and SciCQA (Shen et al., 2024) extend evaluation to schematic diagrams and scientific charts in research papers, but their objectives remain different from diagnosing the internal structure of method figures. Overall, existing chart, flowchart, and scientific-figure benchmarks provide important testbeds, but they do not directly diagnose localized text, component relations, and information flow in real scientific method figures. SciFigBench fills this gap by directly annotating localized text fields and directed component relations in real method figures and constructing diagnostic tasks around them.

2.2 Visual Prompting for Multimodal Reasoning

As VLMs (Radford et al., 2021; Liu et al., 2023; Zhai et al., 2023; Liu et al., 2024; Li et al., 2024a) develop stronger visual reasoning and perception capabilities (Zhang et al., 2026b,a; Li et al., 2026), visual prompting (Zhou et al., 2022; Jia et al., 2022; Shtedritski et al., 2023; Li et al., 2024b, 2025) has been used to improve grounding and region understanding with cues such as color, shape, position, discrete marks, boxes, points, and arrows (Shtedritski et al., 2023; Yang et al., 2023; Cai et al., 2024). These methods mainly target generic object grounding, region understanding, or broad visual enhancement, rather than directed connection reasoning in method figures. SciFigCue targets this structured setting by reusing the automatic annotation pipeline behind SciFigBench to render arrow locations, head–tail directions, and query-relevant target regions as visual prompts for relation understanding and information-flow tracing.

3 SciFigBench

We introduce SciFigBench, a diagnostic benchmark for scientific method figure understanding. It decomposes this capability into a set of QA tasks designed to reveal where model reasoning fails. The detailed construction of SciFigBench is described below.

3.1 Dataset Construction

Figure 3 gives an overview of the SciFigBench construction pipeline, which consists of figure collection, content annotation, and structural annotation. Both annotation types are produced by dedicated automated pipelines and manually verified before being used to construct diagnostic QA examples. This process provides SciFigBench with localized textual content and directed structural relations for real method figures.

Data collection. We collect figures from AI conference papers published in 2024–2025, where method figures often present complex methods in compact visual form. We start from 20,000 paper PDFs, extract candidate primary method figures from figure captions, and discard figures that are not method-centric, lack clear information flow, or contain insufficient textual content. After filtering, 4,352 real scientific method figures constitute our evaluation dataset.

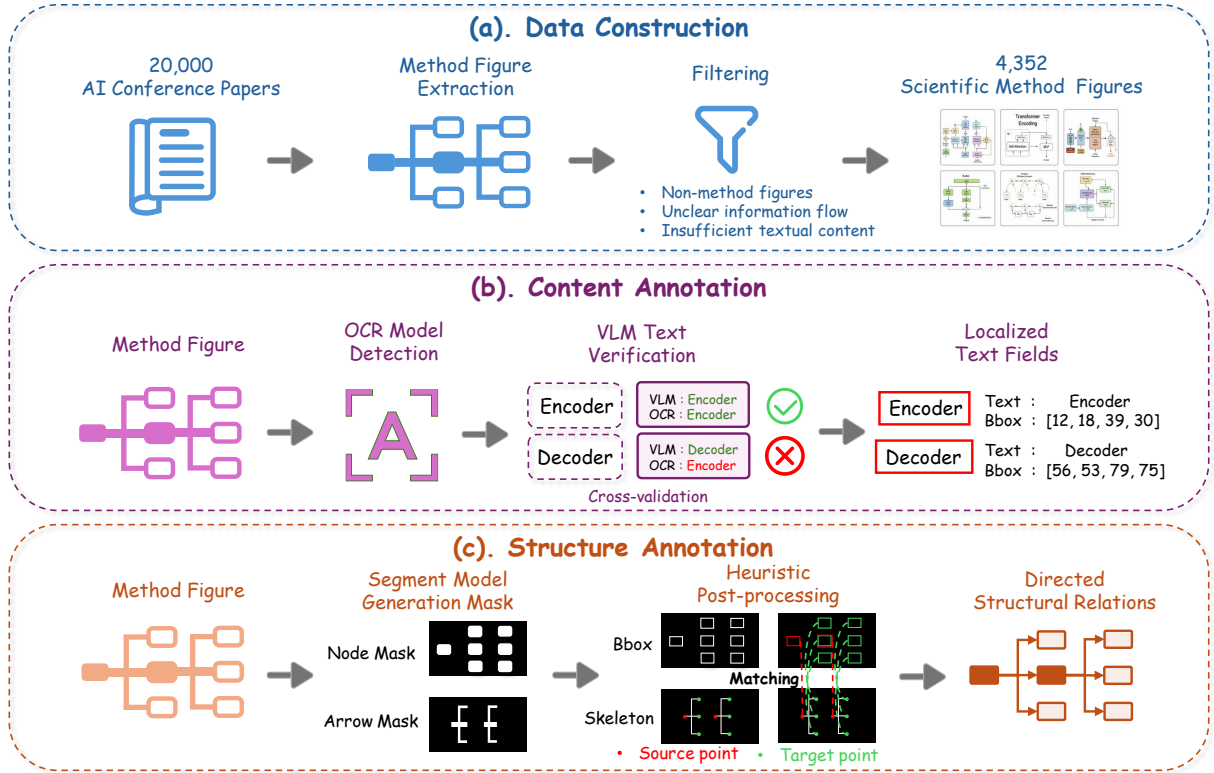


Figure 3: **Overview of the SciFigBench construction pipeline.** (a) Starting from 20,000 AI conference papers, we extract candidate method figures and retain 4,352 scientific method figures after filtering. (b) Content annotation combines OCR detection with VLM-based verification to produce localized text fields. (c) Structural annotation extracts node boxes and head/tail-labeled arrow endpoints through segmentation and post-processing to build directed structural relations. Collectively, these annotations enable the construction of the diagnostic QA tasks in SciFigBench.

Content annotation. The content annotation records localized text fields in each figure, where each field is represented by a text string and its bounding box. We construct this annotation through OCR–VLM cross-validation because method figures often contain small, densely packed text. We first use an OCR model (Cui et al., 2025) to detect text regions and produce OCR hypotheses; a verifier VLM (Bai et al., 2025) then reads each cropped region and selects the more reliable candidate. The content annotation provides text-level supervision for dense content perception. These annotations also support diagnostic tasks that require integrating textual evidence with figure structure.

Structural annotation. The structural annotation captures visual modules, arrow endpoints, and directed connections in each method figure. Each visual module is localized by a node box, and each arrow is represented by a tail endpoint and a head endpoint. Directed connections between nodes are defined by the correspondence between arrow endpoints and node boxes. To construct this annotation,

we first apply a segmentation model (Carion et al., 2026) to generate node and arrow masks, and convert node masks into boxes. The arrow masks are then skeletonized to recover endpoints, and each endpoint neighborhood is cropped for a VLM to determine whether it is the arrow head or tail. Finally, endpoints are associated with node boxes via nearest-boundary matching, yielding candidate directed connections from tail-matched nodes to head-matched nodes. The structural annotation provides structural supervision for visual relation comprehension and information-flow tracing tasks.

All automatically generated content and structural annotations are manually verified by 20 annotators, each receiving over three hours of calibration training before verification. Human verification shows that the automated annotation pipeline achieves an overall F1 of 82.21% across the two annotation types. Detailed verification results are provided in Appendix B. This automatic-first workflow focuses human effort on candidate verification rather than annotation from scratch, reducing anno-

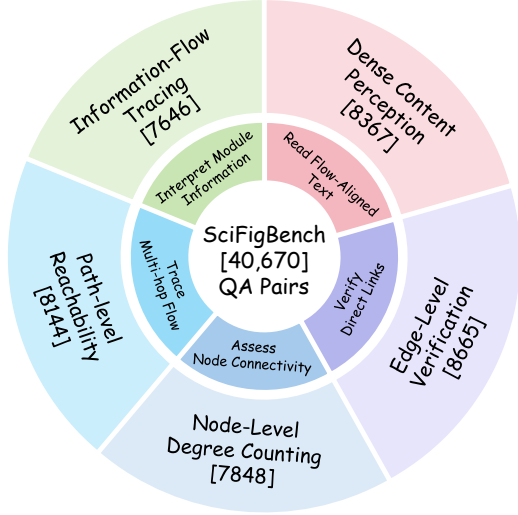


Figure 4: **QA distribution in SciFigBench.** The 40,670 QA pairs are distributed across five diagnostic tasks grouped into three evaluation dimensions: dense content perception, visual relation comprehension, and information-flow tracing.

tation cost. The verified annotations are then used to construct 40,670 QA pairs across five diagnostic QA tasks. Figure 4 shows the distribution of QA pairs across these five tasks. Representative annotation examples are provided in Appendix G.

3.2 Evaluation Tasks

SciFigBench decomposes method-figure understanding into three complementary evaluation perspectives for localizing different capability bottlenecks. These perspectives cover dense content perception, visual relation comprehension, and information-flow tracing. Detailed case construction, case statistics, and task prompts are provided in Appendix A and Appendix H.

3.2.1 Dense Content Perception

Dense content perception evaluates whether VLMs can capture textual cues distributed across method components along an information-flow path. This task reflects how human researchers read method figures when following a method flow: they read component text in context rather than recognize isolated text regions.

Task formulation. Given a method figure, we first select a flow path P from its structural annotation, and then choose an ordered node subsequence $V_{\text{sub}} = (v_1, v_2, \dots, v_K)$ of K text-bearing nodes from P . For each selected node v_i , the aligned text field is denoted as (t_i, b_i) , where t_i is the text string

and b_i is its region box. The task input is the figure with region boxes $(b_1, b_2, \dots, b_K)$ highlighted by a fixed color sequence following the node order in V_{sub} . The model must output the corresponding text list $[t_1, t_2, \dots, t_K]$ in color order. This task requires the model to localize distributed regions, read local text, and preserve the ordering induced by the information flow.

Sampling and metric. For each figure, we sample from the longest flow path to cover a more complete method flow. The path length is limited to $K \leq 6$. When the number of candidate text-bearing nodes exceeds six, we sample uniformly across text-length buckets to cover different reading difficulties. This task yields 8,367 QA pairs with broad coverage across different values of K . The primary metric is position-wise ANLS (Biten et al., 2019) averaged across queried positions.

3.2.2 Visual Relation Comprehension

Visual relation comprehension evaluates whether VLMs can understand arrow-induced structural relations between method components. A central operation in reading method figures is to follow arrows and trace how information moves between components. To evaluate this ability at different levels of structural complexity, we construct three subtasks from the structural annotation: edge-level verification, node-level degree counting, and path-level reachability.

Task formulation. For each method figure, the structural annotation provides a directed graph $G = (V, E)$, where each node $v \in V$ is localized by a corresponding region box and E denotes arrow-induced directed connections. Visual relation queries highlight the region boxes of one or two target nodes on the original figure. The model then reasons about structural relations in G based on the visible arrows. The three subtasks evaluate the model’s ability to determine whether a candidate edge $e = (v_s, v_t)$ belongs to E , predict the in-/out-degree of a target node v , and determine whether v_t is reachable from v_s . The model outputs either a binary decision or a non-negative integer, depending on the query type. All three subtasks are evaluated using accuracy.

Edge-Level Verification. The most local form of relation understanding is determining whether a directed connection exists between two components. Edge-Level Verification evaluates the model’s abil-

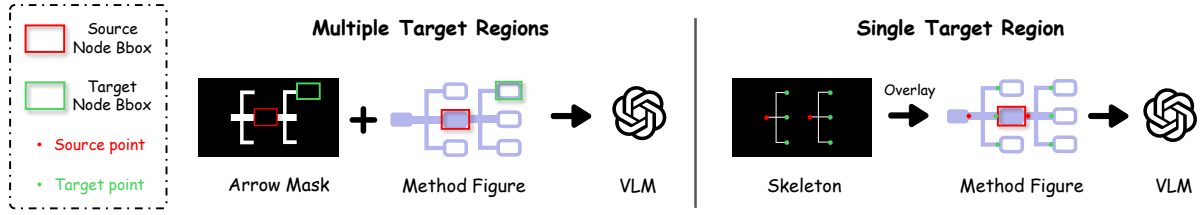


Figure 5: **Two visual-prompt formats in SciFigCue.** Visual cues are extracted by the automated parsing pipeline used in SciFigBench construction and rendered in two formats: an auxiliary arrow-mask view for multi-target queries and an original-figure overlay for single-target queries.

ity to recognize directed relations between two method components. Given a figure where the region boxes of a candidate source node v_s and target node v_t are highlighted in red and blue, respectively, the model must decide whether $(v_s, v_t) \in E$. We balance positive and negative examples and construct hard negatives, yielding 8,665 QA pairs.

Node-Level Degree Counting. Beyond a single edge, the local role of a component also depends on how many connections enter or leave it. Node-Level Degree Counting evaluates the model’s ability to aggregate local connectivity around a target component. Given a figure where the region box of a target node v is highlighted and a query direction $d \in \{\text{in}, \text{out}\}$ is specified, the model must output the number of arrows incident to the target node in that direction. We bucket examples by the target degree to cover diverse local connectivity patterns, yielding 7,848 QA pairs.

Path-Level Reachability. At a higher level, relation understanding requires composing multiple directed connections to determine longer-range dependencies between components. Path-Level Reachability primarily evaluates the model’s ability to reason along directed connections beyond direct adjacency. Given a figure where the region boxes of a source node v_s and a target node v_t are highlighted in red and blue, respectively, the model must determine whether a directed path exists from v_s to v_t . Positive and negative examples are carefully designed to cover diverse reachability patterns, resulting in 8,144 QA pairs.

3.2.3 Information-Flow Tracing

Information-flow tracing further requires models to jointly reason over textual evidence and directed structure. Human researchers often read method figures by anchoring on a named component and tracing information flow along its incoming and

outgoing connections to understand the component’s role in the method flow. To evaluate this ability, we design a neighbor-enumeration QA task that requires named-node localization, directional tracing, and neighbor text recognition without visual box prompts.

Task formulation. Given a method figure, a target node specified by its exact textual label $\langle \text{TARGET} \rangle$, and a query direction $d \in \{\text{in}, \text{out}\}$, the model must return the textual contents of all direct neighboring nodes in the queried direction. For $d = \text{in}$, the answer contains the text from all text-bearing nodes that directly point to the target; for $d = \text{out}$, it contains the text from all text-bearing nodes directly pointed to by the target. The output is an unordered string list $[t_1, t_2, \dots, t_n]$. The target label must be unique within the figure; this constraint focuses the task on information-flow tracing after named-node localization.

Sampling and metric. To construct valid queries, we sample targets only from text-bearing nodes with unique textual labels and create paired in/out queries for each target. During sampling, we match the distribution of candidate-neighbor counts across the in/out directions. The proportion of empty answers is also controlled to reduce potential shortcuts arising from directional bias or empty-answer frequency. The resulting task comprises 7,646 QA pairs. Set-based ANLS* (Peer et al., 2024) is used as the evaluation metric to account for the unordered nature of the answers.

4 SciFigCue

To make fine-grained connection cues in method figures explicit to VLMs, we introduce **SciFigCue**, a training-free input preprocessing module.

Formulation. Given a scientific method figure I and a QA query q , a standard VLM directly predicts an answer a from (I, q) . SciFigCue instead

Model	Size	Dense Content Perception	Edge-Level Verification	Node-Level Degree Counting	Path-Level Reachability	Information-Flow Tracing
RANDOMCHOICE	–	–	50.0	–	50.0	–
<i>Closed-source models</i>						
Gemini-2.5-Flash	–	72.7	66.0	42.3	55.3	42.4
Gemini-3.1-Flash-Lite	–	71.2	<u>67.3</u>	55.8	54.1	46.1
GPT-5-mini	–	51.3	49.8	24.8	49.3	26.0
GPT-5.5	–	<u>76.8</u>	55.8	34.7	51.8	38.6
<i>Open-source models</i>						
InternVL3.5-2B (Wang et al., 2025)	2B	32.5	36.4	31.0	46.4	17.6
InternVL3.5-8B (Wang et al., 2025)	8B	45.9	44.7	35.8	41.5	27.2
InternVL3.5-14B (Wang et al., 2025)	14B	53.7	47.7	35.7	46.3	26.1
InternVL3.5-38B (Wang et al., 2025)	38B	52.2	47.8	40.9	45.4	26.4
LLaVA-OV-1.5-4B (An et al., 2025)	4B	46.0	44.0	33.0	41.8	24.6
LLaVA-OV-1.5-8B (An et al., 2025)	8B	54.8	45.4	34.5	40.2	24.5
Qwen3-VL-4B (Bai et al., 2025)	4B	51.6	48.6	38.9	44.6	27.5
Qwen3-VL-8B (Bai et al., 2025)	8B	55.1	48.9	39.3	46.1	31.3
Qwen3-VL-32B (Bai et al., 2025)	32B	62.6	54.0	46.4	49.4	35.4
Qwen3.5-VL-2B	2B	19.8	46.2	22.0	49.2	16.3
Qwen3.5-VL-4B	4B	45.1	48.8	37.5	46.1	26.3
Qwen3.5-VL-9B	9B	61.2	50.8	41.4	46.3	32.8
Qwen3.5-VL-27B	27B	70.9	59.2	50.9	50.2	<u>43.6</u>
<i>Ours</i>						
SciFigCue (Qwen3.5-VL-9B)	9B	73.0(+11.8)	56.6(+5.8)	44.1(+2.7)	52.0(+5.7)	36.0(+3.2)
SciFigCue (Qwen3.5-VL-27B)	27B	82.4 (+11.5)	70.1 (+10.9)	<u>51.1</u> (+0.2)	56.0 (+5.8)	42.1(-1.5)

Table 1: **Main results on SciFigBench.** All values are percentages; higher is better. Best and second-best results in each column are shown in **bold** and underlined, respectively. Colored parentheses in SciFigCue rows indicate absolute changes over the corresponding base model. Overall, current VLMs remain limited in method figure understanding, while SciFigCue improves performance by making connection cues explicit.

transforms I into an enhanced visual input $\tilde{I}$ by explicitly marking arrow locations, head–tail directions, and query-relevant target regions. The VLM then predicts the answer from $(\tilde{I}, q)$.

Automatic Cue Extraction. SciFigCue reuses the automated parsing pipeline from SciFigBench construction at test time to extract intermediate parsing results for visual prompting. These intermediate results include text or node regions for grounding query targets, arrow masks for isolating connection structure, and head–tail endpoints for marking direction.

Task-Adaptive Visual Prompting. SciFigCue uses two visual-prompt formats depending on whether the query is centered on a single target node or involves multiple target regions. Figure 5 illustrates the overall pipeline for these two prompting formats. For multi-target queries, SciFigCue provides the original figure together with an auxiliary arrow-mask view. The auxiliary view uses the arrow mask extracted by the segmentation model (Carion et al., 2026) as its background and overlays the target regions associated with the current query. This representation reduces visual clutter and makes the connections between target regions more explicit. This format is used

for dense content perception, edge-level verification, and path-level reachability. For single-target queries, SciFigCue overlays visually distinct arrow endpoint cues directly on the original figure, making incoming and outgoing connections around the target node explicit. This format is used for node-level degree counting and information-flow tracing. Additionally, for text-specified targets in information-flow tracing, SciFigCue matches the target name using the content annotation and highlights the corresponding region. SciFigCue changes only the input representation, leaving the answer space, evaluation protocol, and model parameters unchanged.

5 Experiments

5.1 Experimental Setup

We use SciFigBench to evaluate current VLMs on scientific method-figure understanding and to test whether explicit connection cues improve method-figure reasoning. The SciFigBench evaluation comprises 40,670 QA pairs and follows the task inputs and output formats defined in Section 3. In subsequent tables, we abbreviate the five evaluation tasks as DCP, Edge, Degree, Reach., and IFT. We report position-wise ANLS for DCP, set-based ANLS*

for IFT, and accuracy for Edge, Degree, and Reachability. The evaluated model set includes open-source families such as InternVL3.5, LLaVA-OV-1.5, Qwen3.5-VL, and Qwen3-VL, together with proprietary models from Gemini and GPT. Open-source models are evaluated on the full test set, while proprietary models are evaluated on 2,000 QA pairs sampled proportionally by task due to inference costs. Evaluation of open-source models on the same 2,000-QA subset shows only minor performance changes and leaves the overall conclusions unchanged. Detailed subset results are reported in Appendix D.

5.2 Main Results

Overall performance. Table 1 shows that SciFigBench remains challenging for current VLMs in both textual perception and structural reasoning. Among models evaluated without SciFigCue, the best scores are 76.8 on Dense Content Perception and 46.1 on Information-Flow Tracing, indicating substantial room for improvement in dense text perception and text-structure integration. Structural reasoning is even more challenging, with chance accuracy at 50.0 on both Edge-Level Verification and Path-Level Reachability. Many strong models remain near chance on Path-Level Reachability, such as Qwen3.5-VL-27B at 50.2 and Qwen3-VL-32B at 49.4. Node-Level Degree Counting is similarly difficult, with the best result reaching only 55.8. Overall, current VLMs are limited not only in text perception but also in reliably aggregating and composing directed relations in method figures. Such structural reasoning is straightforward for human readers, while current VLMs still struggle to perform it reliably.

Scaling behavior. Model scaling improves text perception more consistently, but does not reliably improve structural reasoning. Across open-source families, larger models tend to perform better on Dense Content Perception and Information-Flow Tracing. Scaling Qwen3.5-VL from 2B to 27B yields gains of 51.1 and 27.3 points on these two tasks, respectively. In contrast, this trend is much weaker for path-level structural reasoning. Across the same scale range, Path-Level Reachability improves by only 1.0 point. This suggests that insufficient model capacity is not the only bottleneck in method-figure understanding. Current models still struggle to jointly reason over arrows, directions, and component layouts.

Method	DCP	Edge	Degree	Reach.	IFT
Qwen3.5-VL-27B	70.9	59.2	50.9	50.2	43.6
AutoParse+Symbolic	96.3	77.1	57.0	65.2	48.4
Verified Oracle	100.0	100.0	100.0	100.0	100.0

Table 2: **Automatic parsing and symbolic reasoning.** AutoParse+Symbolic relies only on outputs from the automated parsing pipeline used in SciFigBench construction, with query regions matched to parsed nodes by geometry. Verified Oracle uses human-verified annotations. Both VLMs and AutoParse+Symbolic remain substantially below the oracle.

SciFigCue gains. The gains from SciFigCue further indicate that this bottleneck is not solely a matter of model scale alone. Without updating model parameters, SciFigCue improves multiple structure-dependent tasks by making arrow locations, directions, and target regions explicit. These gains suggest that current VLMs struggle to reliably extract and exploit fine-grained connection cues from raw method figures. Therefore, the key challenge extends beyond reading figure text to translating arrow-induced visual structure into reliable relational reasoning.

5.3 Diagnostic Analyses

Automatic parsing and symbolic reasoning. To separate structural parsing from downstream graph reasoning, we construct an AutoParse+Symbolic baseline that performs deterministic reasoning over automatically recovered graph structure. This baseline uses only pre-verification parsing outputs, with no access to human-verified annotations or QA answers. As shown in Table 2, AutoParse+Symbolic substantially outperforms Qwen3.5-VL-27B, yet both remain well below the oracle on graph-dependent tasks. The remaining gap between AutoParse+Symbolic and the oracle primarily stems from upstream structure-recovery errors. Missing, spurious, or reversed edges directly affect downstream reasoning tasks. The oracle results indicate that downstream graph reasoning is relatively straightforward once accurate graph structure is available. Combined with the VLM results, these findings identify reliable recovery and use of directed information-flow structure from raw figures as the primary bottleneck.

Fine-grained diagnostic analysis. To probe model behavior beyond aggregate scores, we stratify Dense Content Perception results by the number

Model	$K=1$	$K \geq 2$ Avg.	Gap
Qwen3.5-VL-9B	76.0	45.5	30.5
+ SciFigCue	84.0	61.5	22.5
Qwen3.5-VL-27B	75.9	65.8	10.1
+ SciFigCue	85.2	79.7	5.5

Table 3: **K -wise analysis of Dense Content Perception.** $K \geq 2$ Avg. averages $K = 2, \dots, 6$, and Gap denotes its difference from $K = 1$. SciFigCue consistently narrows the performance gap between the two settings.

Setting	Edge	Degree	Reach.
Raw	50.8	41.4	46.3
Box only	48.3	40.7	47.9
+ Arrow location	53.8	37.8	48.7
+ Randomized arrow	49.9	41.3	46.8
SciFigCue	56.6	44.1	52.0

Table 4: **Controlled cue ablation on Qwen3.5-VL-9B.** Raw denotes the standard SciFigBench input. Within each task, the prompted variants share the same rendering design and differ only in cue content. SciFigCue includes both arrow location and direction. For Degree, Box only strengthens the target highlight without adding arrow cues. Full SciFigCue performs best.

of queried regions K . A clear gap emerges between single- and multi-region reading: the $K = 1$ score exceeds the $K \geq 2$ macro average by 30.5 points for 9B and 10.1 points for 27B. SciFigCue narrows these gaps to 22.5 and 5.5 points for 9B and 27B, respectively. These results further show that current VLMs struggle to extract and exploit arrow-induced information-flow cues from raw method figures. By making these cues explicit, SciFigCue substantially strengthens ordered text perception across multiple regions. Further fine-grained analyses for the remaining tasks are provided in Appendix E.

5.4 SciFigCue Analysis

Controlled cue ablation. Controlled ablations are conducted on Qwen3.5-VL-9B across the Visual Relation Comprehension subtasks, with only the cue content varied and all other settings fixed. As shown in Table 4, full SciFigCue achieves the best performance on all three tasks. The randomized arrow cue preserves the same visual-prompt format but breaks its correspondence with the true arrow structure, reducing performance to near the Raw baseline. This indicates that the gains arise from semantically correct arrow information rather than additional visual salience or arbitrary mark-

Model	Cue	Degree	IFT
Qwen3.5-VL-9B	Auto	44.1	36.0
Qwen3.5-VL-9B	GT	46.1	36.9
Qwen3.5-VL-27B	Auto	51.1	42.1
Qwen3.5-VL-27B	GT	52.7	43.6

Table 5: **Sensitivity to arrow cue quality.** Auto uses automatically parsed cues, while GT uses human-verified cues. GT cues further improve SciFigCue performance, eliminating the IFT degradation observed with Auto cues on Qwen3.5-VL-27B.

ers. Arrow-location cues alone improve Edge and Reachability but degrade Degree, indicating task-dependent sensitivity to cue content. This result suggests that edge and path level reasoning benefits from arrow layout, whereas in/out-degree counting relies more strongly on accurate direction.

Sensitivity to cue quality. We further evaluate a ground-truth (GT) variant that replaces automatically parsed arrow locations and head–tail labels with human-verified cues while keeping the visual-prompt format fixed. This setting isolates cue quality from the prompt format and model parameters. This evaluation focuses on Degree and IFT, which rely on local arrow direction and relation tracing along information flow, respectively. Table 5 shows that GT cues consistently improve performance across both VLMs and tasks. This consistent improvement indicates that SciFigCue remains sensitive to cue fidelity, with more accurate arrow locations and directions providing stronger structural evidence for VLM reasoning.

6 Conclusion

This paper investigates whether current VLMs can reliably understand text, structural connections, and information flow in scientific method figures. We introduce **SciFigBench**, a diagnostic benchmark that evaluates local content perception, structural relation understanding, and information-flow tracing in real method figures. Experiments show that current VLMs not only miss textual evidence but also struggle to aggregate directed relations and trace multi-step dependencies, indicating that reliably recovering and using figure-internal connection structure remains a key bottleneck in method-figure understanding. **SciFigCue** improves structured reasoning without model updates by making fine-grained connection cues explicit, further highlighting reliable connection-structure reasoning as an important direction for future scientific VLMs.

Limitations

SciFigBench provides a diagnostic benchmark for evaluating fine-grained VLM understanding of scientific method figures, but several limitations remain. First, the current dataset is limited to AI conference papers published in 2024–2025 and therefore does not cover the diverse visual conventions, layouts, notations, and arrow semantics found in other scientific domains. We view AI method figures as a practical first step toward broader scientific method-figure understanding; future versions will expand SciFigBench to a wider range of scientific disciplines and adapt SciFigCue to domain-specific visual conventions and arrow semantics. Second, SciFigBench relies on an automated annotation pipeline to generate candidate content and structural annotations, and candidate quality affects annotation cost and scalability. Given the scale of the benchmark and the complexity of scientific method figures, residual annotation errors or ambiguous cases may remain even after human verification. We will continue to refine SciFigBench in future releases based on community feedback. Finally, SciFigCue relies on the same automated parsing pipeline to generate test-time visual cues, making its effectiveness sensitive to upstream structural parsing quality. Further advances in scientific figure parsing may therefore improve SciFigCue performance.

Acknowledgments

This research was supported by the Guangdong Basic and Applied Basic Research Foundation (2026A1515011435), the Fund of the National Natural Science Foundation of China (Grant No. 62576177), Shenzhen Science and Technology Program (QNXMB20250701090801002, JCYJ20250604184027034), the Fundamental Research Funds for the Central Universities 070-63263247, Youth Interdisciplinary Research Project of Nankai University (Grant No. NKQJNC202620) and the Tianjin Key Laboratory of Visual Computing and Intelligent Perception (VCIP). Computation is supported by the Supercomputing Center of Nankai University (NKSC). This work was also supported by the Academy for Advanced Interdisciplinary Studies, Nankai University (AAIS, NKU).

References

- Xiang An, Yin Xie, Kaicheng Yang, Wenkang Zhang, Xiuwei Zhao, Zheng Cheng, Yirui Wang, Songcen Xu, Changrui Chen, Didi Zhu, Chunsheng Wu, Huajie Tan, Chunyuan Li, Jing Yang, Jie Yu, Xiyao Wang, Bin Qin, Yumeng Wang, Zizhen Yan, and 4 others. 2025. Llava-onevision-1.5: Fully open framework for democratized multimodal training. *arXiv preprint arXiv:2509.23661*.
- Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhi-fang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, and 45 others. 2025. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*.
- Ali Furkan Biten, Ruben Tito, Andres Mafla, Lluís Gomez, Marçal Rusinol, Ernest Valveny, CV Jawahar, and Dimosthenis Karatzas. 2019. Scene text visual question answering. In *ICCV*, pages 4291–4301.
- Mu Cai, Haotian Liu, Siva Karthik Mustikovela, Gregory P Meyer, Yuning Chai, Dennis Park, and Yong Jae Lee. 2024. Vip-llava: Making large multimodal models understand arbitrary visual prompts. In *CVPR*, pages 12914–12923. IEEE.
- Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, Jie Lei, Tengyu Ma, Baishan Guo, Arpit Kalla, Markus Marks, Joseph Greer, Meng Wang, Peize Sun, Roman Rädle, and 19 others. 2026. Sam 3: Segment anything with concepts. In *ICLR*, volume 2026, pages 138846–138923.
- Cheng Cui, Ting Sun, Manhui Lin, Tingquan Gao, Yubo Zhang, Jiaxuan Liu, Xueqing Wang, Zelun Zhang, Changda Zhou, Hongen Liu, Yue Zhang, Wenyu Lv, Kui Huang, Yichao Zhang, Jing Zhang, Jun Zhang, Yi Liu, Dianhai Yu, and Yanjun Ma. 2025. Paddleocr 3.0 technical report. *arXiv preprint arXiv:2507.05595*.
- Aniketh Garikaparthi, Manasi Patwardhan, Aditya Sanjiv Kanade, Aman Hassan, Lovekesh Vig, and Arman Cohan. 2025. Mir: Methodology inspiration retrieval for scientific research problems. In *ACL Main*, pages 28614–28659.
- Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. 2022. Visual prompt tuning. In *ECCV*, pages 709–727. Springer.
- Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, and Chunyuan Li. 2024a. Llava-onevision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*.

- Danyang Li, Tianhao Wu, Bin Lin, Zhenyuan Chen, Yang Zhang, Yuxuan Li, Ming-Ming Cheng, and Xiang Li. 2026. Wow-seg: A word-free open world segmentation model. In *ICLR*, volume 2026, pages 155933–155958.
- Zheng Li, Xiang Li, Xinyi Fu, Xin Zhang, Weiqiang Wang, Shuo Chen, and Jian Yang. 2024b. Promptkd: Unsupervised prompt distillation for vision-language models. In *CVPR*, pages 26607–26616. IEEE.
- Zheng Li, Yibing Song, Ming-Ming Cheng, Xiang Li, and Jian Yang. 2025. Advancing textual prompt learning with anchored attributes. In *ICCV*, pages 3618–3627. IEEE.
- Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. 2024. Improved baselines with visual instruction tuning. In *CVPR*, pages 26286–26296. IEEE.
- Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual instruction tuning. In *NeurIPS*, volume 36, pages 34892–34916.
- Ahmed Masry, Do Xuan Long, Jia Qing Tan, Shafiq Joty, and Enamul Hoque. 2022. Chartqa: A benchmark for question answering about charts with visual and logical reasoning. In *ACL Findings*, pages 2263–2279.
- Huitong Pan, Qi Zhang, Cornelia Caragea, Edward Dragut, and Longin Jan Latecki. 2024. Flowlearn: Evaluating large vision-language models on flowchart understanding. In *ECAI*, pages 73–80.
- David Peer, Philemon Schöpf, Volckmar Nebendahl, Alexander Rietzler, and Sebastian Stabinger. 2024. Anls*—a universal document processing metric for generative large language models. *arXiv preprint arXiv:2402.03848*.
- Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *ICML*, pages 8748–8763. PMLR.
- Lingdong Shen, Qigqi, Kun Ding, Gaofeng Meng, and Shiming Xiang. 2024. Rethinking comprehensive benchmark for chart understanding: A perspective from scientific literature. *arXiv preprint arXiv:2412.12150*.
- Aleksandar Shtedritski, Christian Rupprecht, and Andrea Vedaldi. 2023. What does clip know about a red circle? visual prompt engineering for vlms. In *ICCV*, pages 11953–11963. IEEE.
- Shubhankar Singh, Purvi Chaurasia, Yerram Varun, Pranshu Pandya, Vatsal Gupta, Vivek Gupta, and Dan Roth. 2024. Flowvqa: Mapping multimodal logic in visual question answering with flowcharts. In *ACL Findings*, pages 1330–1350.
- Manan Suri, Puneet Mathur, Nedim Lipka, Franck Dernoncourt, Ryan A Rossi, Vivek Gupta, and Dinesh Manocha. 2025. Follow the flow: Fine-grained flowchart attribution with neurosymbolic agents. In *EMNLP Main*, pages 22496–22519.
- Simon Tannert, Marcelo G. Feighelstein, Jasmina Bogojeska, Joseph Shtok, Assaf Arbelle, Peter W. J. Staar, Anika Schumann, Jonas Kuhn, and Leonid Karlinsky. 2023. Flowchartqa: The first large-scale benchmark for reasoning over flowcharts. In *Proceedings of the 1st Workshop on Linguistic Insights from and for Multimodal Language Processing*, pages 34–46.
- Qingyun Wang, Doug Downey, Heng Ji, and Tom Hope. 2024. Scimon: Scientific inspiration machines optimized for novelty. In *ACL Main*, pages 279–299.
- Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, Zhaokai Wang, Zhe Chen, Hongjie Zhang, Ganlin Yang, Haomin Wang, Qi Wei, Jinhui Yin, Wenhao Li, Erfei Cui, and 56 others. 2025. Internvl3.5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*.
- Jingxuan Wei, Nan Xu, Junnan Zhu, Haoyanni, Gaowei Wu, Qi Chen, Bihui Yu, and Lei Wang. 2025. Chartmind: A comprehensive benchmark for complex real-world multimodal chart question answering. In *EMNLP Main*, pages 4555–4569.
- Jianwei Yang, Hao Zhang, Feng Li, Xueyan Zou, Chunyuan Li, and Jianfeng Gao. 2023. Set-of-mark prompting unleashes extraordinary visual grounding in gpt-4v. *arXiv preprint arXiv:2310.11441*.
- Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. 2023. Sigmoid loss for language image pre-training. In *ICCV*, pages 11941–11952. IEEE.
- Xu Zhang, Danyang Li, Xiaohang Dong, Tianhao Wu, Hualong Yu, Jianye Wang, Qicheng Li, and Xiang Li. 2026a. Unichange: Unifying change detection with multimodal large language model. In *CVPR*, pages 42169–42179.
- Yang Zhang, Danyang Li, Yuxuan Li, Xin Zhang, Tianyu Xie, Mingming Cheng, and Xiang Li. 2026b. Crystal: Spontaneous emergence of visual latents in mllms. *arXiv preprint arXiv:2602.20980*.
- Yu Zhang, Xiusi Chen, Bowen Jin, Sheng Wang, Shuiwang Ji, Wei Wang, and Jiawei Han. 2024. A comprehensive survey of scientific large language models and their applications in scientific discovery. In *EMNLP Main*, pages 8783–8817.
- Penghai Zhao, Jinyu Tian, Qinghua Xing, Xin Zhang, Zheng Li, Jianjun Qian, Ming-Ming Cheng, and Xiang Li. 2026. Naipv2: Debaised pairwise learning for efficient paper quality estimation. In *ICLR*, volume 2026, pages 144257–144279.

Yilun Zhao, Weiyuan Chen, Zhijian Xu, Manasi Patwardhan, Chengye Wang, Yixin Liu, Lovekesh Vig, and Arman Cohan. 2025a. Abgen: Evaluating large language models in ablation study design and evaluation for scientific research. In *ACL Main*, pages 12479–12491.

Yilun Zhao, Chengye Wang, Chuhan Li, and Arman Cohan. 2025b. Can multimodal foundation models understand schematic diagrams? an empirical study on information-seeking qa over scientific papers. In *ACL Findings*, pages 18598–18631.

Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. 2022. Learning to prompt for vision-language models. *IJCV*, 130(9):2337–2348.

A QA Construction Details

This section provides additional details on SciFig-Bench QA construction. We summarize the case design and statistics for each task.

Dense Content Perception. Dense Content Perception is constructed as a unified flow-path OCR query, where the model reads K highlighted text regions in information-flow order. When $K = 1$, the task reduces to single-region OCR; when $K \geq 2$, it evaluates ordered reading across multiple method components. We retain a substantial fraction of $K = 1$ examples because localized text reading provides the foundation for ordered reading across multiple regions. Table 6 summarizes the distribution over sequence length K .

K	1	2	3	4	5	6	Total
Count	4,352	730	1,014	811	623	837	8,367

Table 6: **Dense Content Perception case distribution.** Examples are grouped by the number of highlighted text regions K . $K = 1$ denotes the single-region special case, while $K \geq 2$ denotes flow-path multi-region reading.

Edge-Level Verification. Edge-Level Verification balances positive and negative examples and includes hard negatives. Reverse cases select two nodes connected by a true edge, but the true edge direction is opposite to the red-to-blue query direction. Two-hop cases select node pairs that are connected through a two-hop path but do not have a direct edge. Sibling-source cases select node pairs that share the same source, and sibling-target cases select pairs that share the same target. Spatial-close cases select node pairs that are spatially close but not directly connected in the graph. Together, these

Case	Count
Positive	4,341
Negative / easy	2,152
Negative / reverse	493
Negative / sibling source	317
Negative / sibling target	397
Negative / spatial close	523
Negative / two-hop	442
Total	8,665

Table 7: **Edge-Level Verification case distribution.** Positive examples contain a directed edge from the red-boxed node to the blue-boxed node. Negative examples are grouped by construction type, including easy negatives and hard negatives.

Direction	0	1	2	3	4+	Total
In	746	1,141	1,152	635	279	3,953
Out	717	1,335	1,053	556	234	3,895
Total	1,463	2,476	2,205	1,191	513	7,848

Table 8: **Node-Level Degree Counting case distribution.** Examples are grouped by query direction and edge-count bucket, where 0 denotes no edge in the queried direction and 4+ denotes denser local connectivity.

hard cases prevent models from relying on simple spatial-proximity or connectivity priors, and instead require them to judge the directionality and directness of visible connections. Table 7 reports the resulting case distribution.

Node-Level Degree Counting. Node-Level Degree Counting queries the in- or out-degree of a target node. We count edges rather than distinct neighboring nodes because connections in method figures usually express specific information transfer or module dependency. Parallel edges and self-loops are counted according to the task definition to preserve fine-grained local connectivity. Examples are stratified by in/out query direction. The 0 bucket indicates no edge in the queried direction, although edges may exist in the opposite direction. Sampling emphasizes the more common 1-, 2-, and 3-degree cases in method figures, while retaining 4+ cases to evaluate denser local connectivity. This sampling design covers absent, common low-degree, and denser local connectivity patterns, evaluating whether models can both detect the absence of directional edges and aggregate existing connections. Table 8 gives the distribution by query direction and target count bucket.

Path-Level Reachability. Path-Level Reachability asks whether a directed path exists from a

Case	Count
Positive / length=1 fallback	162
Positive / length=2	1,782
Positive / length=3	1,173
Positive / length=4	710
Positive / length=5+	409
Negative / easy	1,753
Negative / reverse path	773
Negative / sibling source	329
Negative / sibling target	459
Negative / spatial close	594
Total	8,144

Table 9: **Path-Level Reachability case distribution.** Positive examples are grouped by shortest directed path length, and negative examples are grouped by construction type. Length-one positives are used only as fallbacks for papers without any multi-hop positive path.

highlighted source node to a highlighted target node. Positive examples preferentially select node pairs with a shortest directed path of at least two edges, reducing overlap with direct-edge verification. Length-one direct-edge positives are retained as fallbacks only when a paper contains no positive path of length at least two. Consequently, 4,074 of the 4,236 positive examples require multi-hop reasoning, with only 162 retained as length-one fallbacks. Easy negatives select node pairs that are unreachable in the queried direction and do not fall into a specific hard-negative type. Reverse-path negatives select node pairs where the target node can reach the source node but the source node cannot reach the target node. Sibling-source negatives select node pairs that share the same source, and sibling-target negatives select pairs that share the same target. Spatial-close negatives select node pairs that are spatially close but not reachable in the queried direction. Together, these cases require models to distinguish true directed reachability from reverse reachability, shared local connectivity, and visual proximity. Table 9 reports the resulting case distribution.

Information-Flow Tracing. Information-Flow Tracing asks the model to list the text labels of direct incoming or outgoing neighbors of a named target node. To avoid ambiguous references, we select only target nodes with unique textual labels within the figure. Each eligible target yields a pair of incoming and outgoing queries, testing the same local structure in both information-flow directions. We group targets as in-only, out-only, or both according to whether they have incoming neighbors,

Target type	Query direction	Answer type	Count
Both	Incoming	Non-empty	2,485
Both	Outgoing	Non-empty	2,485
In only	Incoming	Non-empty	669
In only	Outgoing	Empty	669
Out only	Incoming	Empty	669
Out only	Outgoing	Non-empty	669
Total			7,646

Table 10: **Information-Flow Tracing case distribution.** Counts are reported at the query level after each eligible target is expanded into one incoming and one outgoing query. Both-type targets produce two non-empty queries, whereas in-only and out-only targets produce one empty query in the missing direction.

Component	Precision	Recall	F1
OCR	95.46	95.72	95.59
Node segmentation	57.01	71.80	63.56
Arrow endpoint	67.78	87.07	76.23
Overall	76.96	88.23	82.21

Table 11: **Quality of automatic annotations before human verification.** Primary metrics use the matching criteria defined above, with additional semantic correctness requirements for OCR and arrow endpoints. The automatic pipeline achieves an overall primary F1 of 82.21%.

outgoing neighbors, or both. For in-only and out-only targets, the missing direction yields an empty answer, testing whether models can recognize the absence of direct neighbors. Table 10 lists the resulting query-level distribution by target type, query direction, and answer type.

B Automatic Annotation Quality

We evaluate automatic annotation quality over all 4,352 figures in SciFigBench by comparing pre-verification outputs with human-verified annotations. OCR and node boxes are matched at IoU ≥ 0.5 , while arrow endpoints are matched within a 15-pixel distance threshold. Primary correctness additionally requires exact normalized text agreement for OCR and the correct head/tail type for arrow endpoints.

Overall annotation quality. Table 11 reports primary precision, recall, and F1 for OCR, node segmentation, and arrow endpoint annotation. The automatic annotations achieve an overall primary F1 of 82.21%. OCR is relatively reliable, while structural annotations remain substantially more challenging.

Arrow endpoint diagnostic	Value
Endpoint match precision	73.49
Endpoint match recall	94.41
Endpoint match F1	82.65
Head/tail accuracy	92.23
Nearest-node accuracy	71.33
Mean endpoint distance	0.58 px

Table 12: **Diagnostics of automatic arrow endpoint parsing.** Endpoint matching uses a 15-pixel distance threshold. Head/tail and nearest-node accuracy evaluate directional labeling and endpoint-to-node association on matched endpoints, respectively. All matching and accuracy values are percentages.

Arrow endpoint diagnostics. Because arrow cues are an important component of the structural prompts used by SciFigCue, we further analyze arrow endpoint parsing errors. As shown in Table 12, matched endpoints are usually assigned the correct head/tail type, whereas associating them with the correct nodes remains more challenging. Errors in node segmentation and endpoint association can propagate to directed-relation recovery, since valid edges depend on correctly associating endpoints with their corresponding nodes. Thus, structural parsing rather than OCR remains the main bottleneck in the automated pipeline, consistent with the observed sensitivity of SciFigCue to upstream cue quality.

C Implementation Details

The main experiments are conducted on eight NVIDIA A40 GPUs, while additional analyses and ablations are run on eight NVIDIA RTX 3090 GPUs. For open-source models, we set the temperature to 0 and the maximum generation length to 2,048 tokens, with explicit chain-of-thought (CoT) or thinking modes disabled. Proprietary models use the same maximum generation length, with additional reasoning disabled or minimized whenever supported by the corresponding API. Specifically, GPT-5.5 uses a maximum generation length of 2,048 tokens with additional reasoning disabled. Temperature is omitted due to API constraints, and no explicit CoT is enabled. GPT-5-mini also uses a maximum generation length of 2,048 tokens with the lowest supported reasoning setting. Temperature is omitted, and no explicit CoT is enabled. Gemini-2.5-Flash uses a maximum generation length of 2,048 tokens and a temperature of 0, with thinking disabled. Gemini-3.1-Flash-Lite uses the same maximum generation length and tem-

perature. Since its API does not support disabling thinking entirely, we use the lowest available thinking setting.

D Subset Evaluation

To enable a fair comparison with proprietary models, we evaluate all open-source models on the same 2,000-QA subset used for GPT and Gemini models. Table 13 reports the per-model results for both open- and closed-source models on this shared subset. As summarized in Table 14, weighted-mean performance across the 13 open-source models changes by only +0.9 percentage points on average relative to the full test set, indicating that evaluation on the shared subset does not materially alter the overall comparison.

E Task-Specific Diagnostic Analysis

This section further analyzes model performance across task-specific subgroups to reveal fine-grained behaviors hidden by aggregate results. The representative proprietary model, GPT-5.5, uses the same 2,000-QA evaluation subset, while the remaining models use the full test set.

Dense Content Perception. We break down Dense Content Perception by the number of queried regions K to examine the transition from localized reading to ordered reading across multiple regions. As shown in panel (a) of Table 15, most models exhibit a clear performance gap between $K = 1$ and $K \geq 2$. SciFigCue consistently improves Qwen3.5-VL across all $K \geq 2$ settings, further indicating that explicit information-flow cues facilitate ordered reading across distributed regions.

Edge-Level Verification. Panel (b) of Table 15 breaks down Edge-Level Verification by answer label and representative negative subtypes. Several models exhibit substantial positive-negative performance imbalance, indicating that aggregate accuracy can obscure answer bias. Two-hop negatives further expose confusion between direct and indirect relations; for example, Qwen3.5-VL-9B drops from 93.7 on easy negatives to 44.8 on two-hop negatives.

Node-Level Degree Counting. Panel (c) of Table 15 stratifies Node-Level Degree Counting by the degree count in the queried direction. Performance degrades substantially as local connectivity becomes denser. For example, Qwen3.5-VL-27B scores 79.3 at degree 1 but only 14.2 at degree

Model	Size	Dense Content Perception	Edge-Level Verification	Node-Level Degree Counting	Path-Level Reachability	Information-Flow Tracing
RANDOMCHOICE	–	–	50.0	–	50.0	–
<i>Closed-source models</i>						
Gemini-2.5-Flash	–	72.7	66.0	42.3	55.3	42.4
Gemini-3.1-Flash-Lite	–	71.2	67.3	55.8	<u>54.1</u>	<u>46.1</u>
GPT-5-mini	–	51.3	49.8	24.8	49.3	26.0
GPT-5.5	–	76.8	55.8	34.7	51.8	38.6
<i>Open-source models</i>						
InternVL3.5-2B (Wang et al., 2025)	2B	35.2	37.7	31.0	48.1	20.2
InternVL3.5-8B (Wang et al., 2025)	8B	49.4	44.9	38.0	41.5	29.1
InternVL3.5-14B (Wang et al., 2025)	14B	53.5	47.2	35.4	47.0	29.3
InternVL3.5-38B (Wang et al., 2025)	38B	55.3	51.7	46.0	41.9	26.2
LLaVA-OV-1.5-4B (An et al., 2025)	4B	48.2	44.7	35.4	38.5	28.4
LLaVA-OV-1.5-8B (An et al., 2025)	8B	56.7	46.0	32.8	44.2	28.1
Qwen3-VL-4B (Bai et al., 2025)	4B	53.0	50.0	38.0	44.2	28.9
Qwen3-VL-8B (Bai et al., 2025)	8B	56.4	50.5	38.0	46.3	32.8
Qwen3-VL-32B (Bai et al., 2025)	32B	64.8	56.8	43.8	47.3	36.1
Qwen3.5-VL-2B	2B	21.8	45.8	21.9	48.6	18.2
Qwen3.5-VL-4B	4B	45.4	49.2	37.6	48.4	27.5
Qwen3.5-VL-9B	9B	58.5	53.8	39.8	45.6	38.5
Qwen3.5-VL-27B	27B	<u>72.8</u>	59.8	<u>48.9</u>	49.1	49.1

Table 13: **Results on the shared 2,000-QA evaluation subset.** All values are percentages, using the same metrics as Table 1. All closed- and open-source models are evaluated on the same subset. Best and second-best results in each column are shown in **bold** and underlined, respectively.

Group	Δ DCP	Δ Edge	Δ Degree	Δ Reach.	Δ IFT	Δ Mean
Open-source	+1.5	+1.2	-0.1	-0.2	+2.5	+0.9

Table 14: **Average changes from full-test to subset evaluation.** Values are percentage-point differences computed as subset score minus full-test score. Δ Mean denotes the corresponding change in the weighted aggregate across tasks.

4+, revealing dense-connectivity aggregation as a major bottleneck.

Path-Level Reachability. Panel (d) of Table 15 analyzes Path-Level Reachability by answer label and shortest-path length for positive examples. The results reveal substantial answer bias rather than a simple monotonic trend with path length. For example, Qwen3.5-VL-27B scores 77.0 and 25.5 on negative and positive examples, respectively, while SciFigCue changes these scores to 62.7 and 49.7. This contrast shows that aggregate reachability accuracy can conceal strong answer-class preferences.

Information-Flow Tracing. Panel (e) of Table 15 stratifies Information-Flow Tracing by ground-truth neighbor count to examine empty-answer bias. Several models show large discrepancies between zero-neighbor and non-empty cases, revealing empty-answer bias hidden by aggregate scores. For example, Qwen3.5-VL-9B scores

68.0 on zero-neighbor cases but only 24.3 on one-neighbor cases. SciFigCue raises the one-neighbor score to 31.1 and substantially narrows the gap between zero- and one-neighbor cases.

F Color Robustness

Because query-box colors may interact with colors already present in method figures, we further evaluate the robustness of SciFigBench to different box-color schemes. Prior work has shown that visual-prompt color can affect VLM behavior (Shtedritski et al., 2023). We therefore test three perceptually distinct schemes: the original red/blue, green/purple, and cyan/orange palettes. The evaluation covers three VLMs and the full QA sets of the three box-grounded structural tasks: Edge-Level Verification, Node-Level Degree Counting, and Path-Level Reachability. Although absolute scores vary across color schemes, the weighted-score ranking remains unchanged in all settings: Qwen3.5-VL-9B > InternVL3.5-14B > LLaVA-OV-1.5-8B. Thus, changing query-box colors does not alter the model-level comparison, suggesting that the model-level conclusions are not tied to the original red/blue palette.

Local color contrast. We further examine whether query boxes are difficult to distinguish from local figure backgrounds. We use CIELAB ΔE_{76} to measure contrast between box boundaries

(a) Dense Content Perception						
Model	$K=1$	$K=2$	$K=3$	$K=4$	$K=5$	$K=6$
GPT-5.5	83.4	68.4	65.6	75.5	68.2	71.6
Qwen3-VL-32B	69.8	44.4	47.8	54.1	63.7	66.5
Qwen3.5-VL-9B	76.0	47.1	41.9	42.8	48.9	46.6
+ SciFigCue	84.0	66.7	56.6	59.0	64.3	61.1
Qwen3.5-VL-27B	75.9	58.0	60.9	65.4	73.6	71.2
+ SciFigCue	85.2	79.9	76.5	79.6	82.3	80.1

(b) Edge-Level Verification					
Model	Neg.	Pos.	Easy	Reverse	Two-Hop
GPT-5.5	54.7	57.0	64.7	35.3	46.7
Qwen3-VL-32B	87.9	20.2	96.1	94.5	43.7
Qwen3.5-VL-9B	86.6	15.1	93.7	95.1	44.8
+ SciFigCue	64.4	49.0	73.7	81.5	15.2
Qwen3.5-VL-27B	96.4	22.2	98.8	96.6	90.0
+ SciFigCue	87.9	52.3	95.4	88.8	62.4

(c) Node-Level Degree Counting					
Model	0	1	2	3	4+
GPT-5.5	36.5	44.2	26.3	33.3	22.2
Qwen3-VL-32B	50.1	73.5	33.1	26.3	9.0
Qwen3.5-VL-9B	49.8	34.4	62.8	18.6	13.1
+ SciFigCue	41.2	47.7	66.1	14.3	9.9
Qwen3.5-VL-27B	43.3	79.3	47.2	24.2	14.2
+ SciFigCue	44.4	80.6	42.2	30.0	14.2

(d) Path-Level Reachability						
Model	Neg.	Pos.	$L=2$	$L=3$	$L=4$	$L \geq 5$
GPT-5.5	21.7	79.6	83.1	74.4	79.6	78.6
Qwen3-VL-32B	80.2	20.9	20.0	22.2	23.5	20.5
Qwen3.5-VL-9B	74.9	19.9	18.2	21.2	23.8	19.6
+ SciFigCue	37.8	65.0	62.2	66.4	68.3	71.6
Qwen3.5-VL-27B	77.0	25.5	23.5	27.8	28.6	27.9
+ SciFigCue	62.7	49.7	48.5	51.7	52.0	49.1

(e) Information-Flow Tracing				
Model	0	1	2-3	4+
GPT-5.5	12.8	43.8	48.4	20.0
Qwen3-VL-32B	34.3	34.8	39.0	33.2
Qwen3.5-VL-9B	68.0	24.3	30.1	23.0
+ SciFigCue	57.4	31.1	34.0	24.2
Qwen3.5-VL-27B	42.2	43.9	44.9	37.0
+ SciFigCue	21.4	47.0	45.8	36.9

Table 15: **Task-specific diagnostic analysis.** All values are percentages. **(a)** DCP is grouped by sequence length K . **(b)** Edge reports answer labels and representative negative subtypes. **(c)** Degree is grouped by the degree count in the queried direction. **(d)** Reach. reports answer labels and shortest-path length L for positive examples; rare length-one fallbacks are omitted from the L breakdown. **(e)** IFT is grouped by the number of ground-truth neighbors. GPT-5.5 uses the 2,000-QA subset, while the remaining models use the full test set. These breakdowns reveal distributed-reading difficulty, direct/indirect relation confusion, dense-connectivity failures, and answer biases.

and figure backgrounds, treating boxes with a median boundary contrast below 20 as low-contrast. As shown in Table 17, low-contrast cases are negligible across all three schemes, indicating that query boxes do not systematically blend into figure backgrounds. Moreover, color contrast and model performance are not monotonic: green/purple has lower median contrast than red/blue but yields higher weighted scores for all three models. We therefore do not claim color invariance; rather, these results indicate that the main conclusions are not driven by systematic color interference or a single red/blue convention.

G Dataset Annotation Examples

This section presents representative annotation examples from SciFigBench to make the annotation format more concrete. Each example shows both the content annotation and the structural annotation, corresponding to textual content and directed structural relations in the figure. The content annotation localizes text regions, supporting Dense Content Perception and QA construction that requires textual evidence. The structural annotation marks component regions, arrow endpoints, and directed connections between components, supporting relation verification, degree counting, reachability, and information-flow tracing. In the visualizations,

Model	Scheme	Edge	Degree	Reach.	Weighted	Δ Weighted
Qwen3.5-VL-9B	Red/blue	50.8	41.4	46.3	46.3	+0.0
	Green/purple	54.7	40.3	52.3	49.3	+3.0
	Cyan/orange	50.8	36.5	49.4	45.8	-0.5
InternVL3.5-14B	Red/blue	47.7	35.7	46.3	43.4	+0.0
	Green/purple	51.5	35.3	49.2	45.6	+2.2
	Cyan/orange	48.4	34.5	47.4	43.7	+0.3
LLaVA-OV-1.5-8B	Red/blue	45.4	34.5	40.2	40.2	+0.0
	Green/purple	48.7	34.2	45.0	42.9	+2.7
	Cyan/orange	48.3	33.0	44.5	42.2	+2.0

Table 16: **Robustness to query-box color schemes.** All values are percentages. Weighted denotes the task-size-weighted aggregate across Edge, Degree, and Reach., and Δ Weighted is the absolute change relative to the original red/blue scheme.

Scheme	Boxes	Low-contrast	Fraction	Median ΔE	10th pct. ΔE
Red/blue	41,466	0	0.0000%	113.37	107.54
Green/purple	41,466	0	0.0000%	96.76	96.17
Cyan/orange	41,466	4	0.0096%	59.17	56.00

Table 17: **Local visibility of query boxes under different color schemes.** Larger ΔE_{76} indicates stronger local box/background contrast. The 10th percentile summarizes lower-visibility cases.

green arrow endpoints indicate arrow tails, while red arrow endpoints indicate arrow heads. These examples illustrate the dense text, free-form layouts, and arrow-based connections common in real method figures, as well as how SciFigBench QA tasks are grounded in fine-grained annotations.

H Task Prompts

We use fixed task prompts for all examples within each task type, except that Information-Flow Tracing replaces the {target} placeholder with the target node text. The figures below show the prompts in the form used for evaluation.

I Visual Analysis

This section complements the aggregate evaluation with representative visual cases, illustrating typical model errors across all five tasks and the remaining failure modes of SciFigCue. These cases make the specific failure sources hidden by aggregate scores more explicit and further illustrate which errors can be mitigated by explicit structural cues and which remain unresolved.

Task-level qualitative cases. We present a representative failure case for each of the five tasks in a separate figure so that fine-grained textual and structural details remain legible. The selected cases include incorrect ordering in multi-region text reading, misinterpretation of arrow direction in a reverse-edge case, confusion between incoming

and outgoing degree, failure to compose multi-hop directed relations, and incorrect information-flow tracing after locating the target node.

SciFigCue failure modes. Although SciFigCue improves multiple structure-dependent tasks, its effectiveness remains limited by upstream cue fidelity and the model’s residual structural reasoning ability. Inspection of unsuccessful cases reveals two major sources of failure: parser-induced cue errors and residual reasoning errors despite correct structural cues. Parser-induced errors, including missed or spurious arrows, incorrect endpoint attachment, and erroneous head/tail labels, can produce incomplete or misleading structural cues. In dense regions, incorrect overlays may also obscure valid arrow evidence, amplifying upstream parsing errors. Figure 19 illustrates this failure: the model answers the canonical input correctly, but misparsed overlaid cues lead to an incorrect SciFigCue prediction. A second source of failure is residual model reasoning: even when the structural cues are correct, the model may still miscount connections, confuse direct and indirect relations, or fail to compose multi-hop paths. Figure 20 shows such a two-hop hard case with correct cues, where the model still mistakes an indirect connection for a direct edge. These cases align with the cue-quality sensitivity analysis, showing that SciFigCue remains limited by both automatic parsing errors and residual VLM reasoning failures.

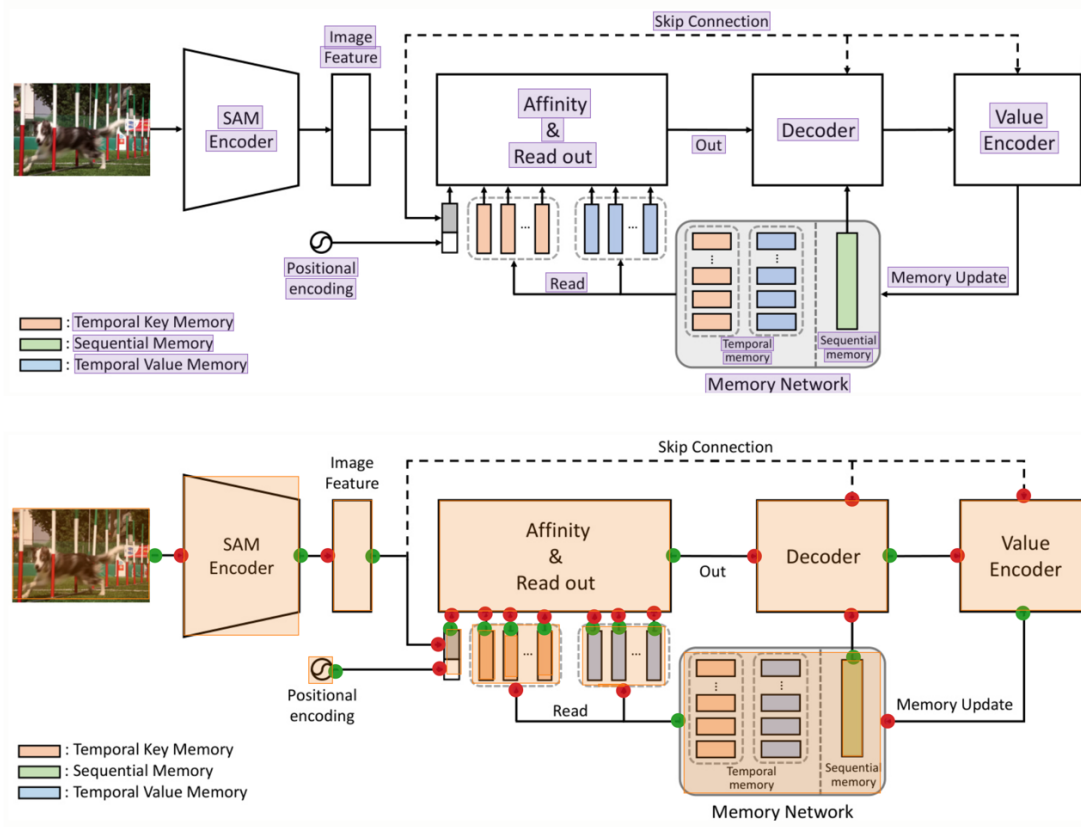


Figure 6: **Dataset annotation example 1.** Content and structural annotations for a real method figure.

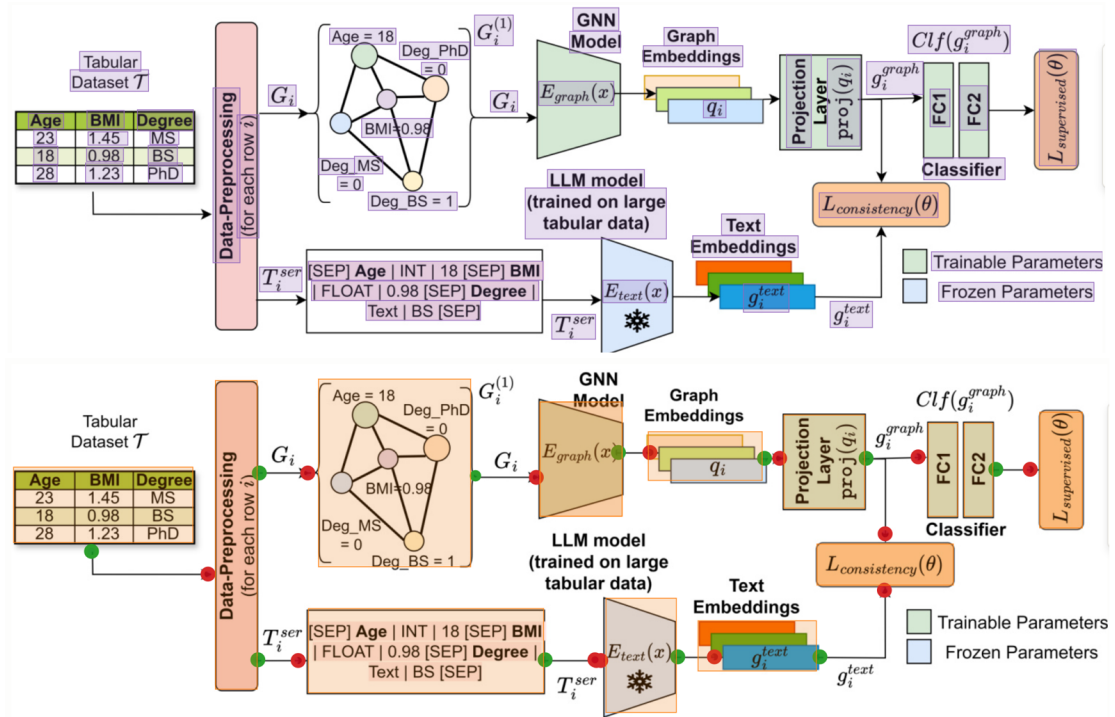


Figure 7: **Dataset annotation example 2.** Content and structural annotations for a real method figure.

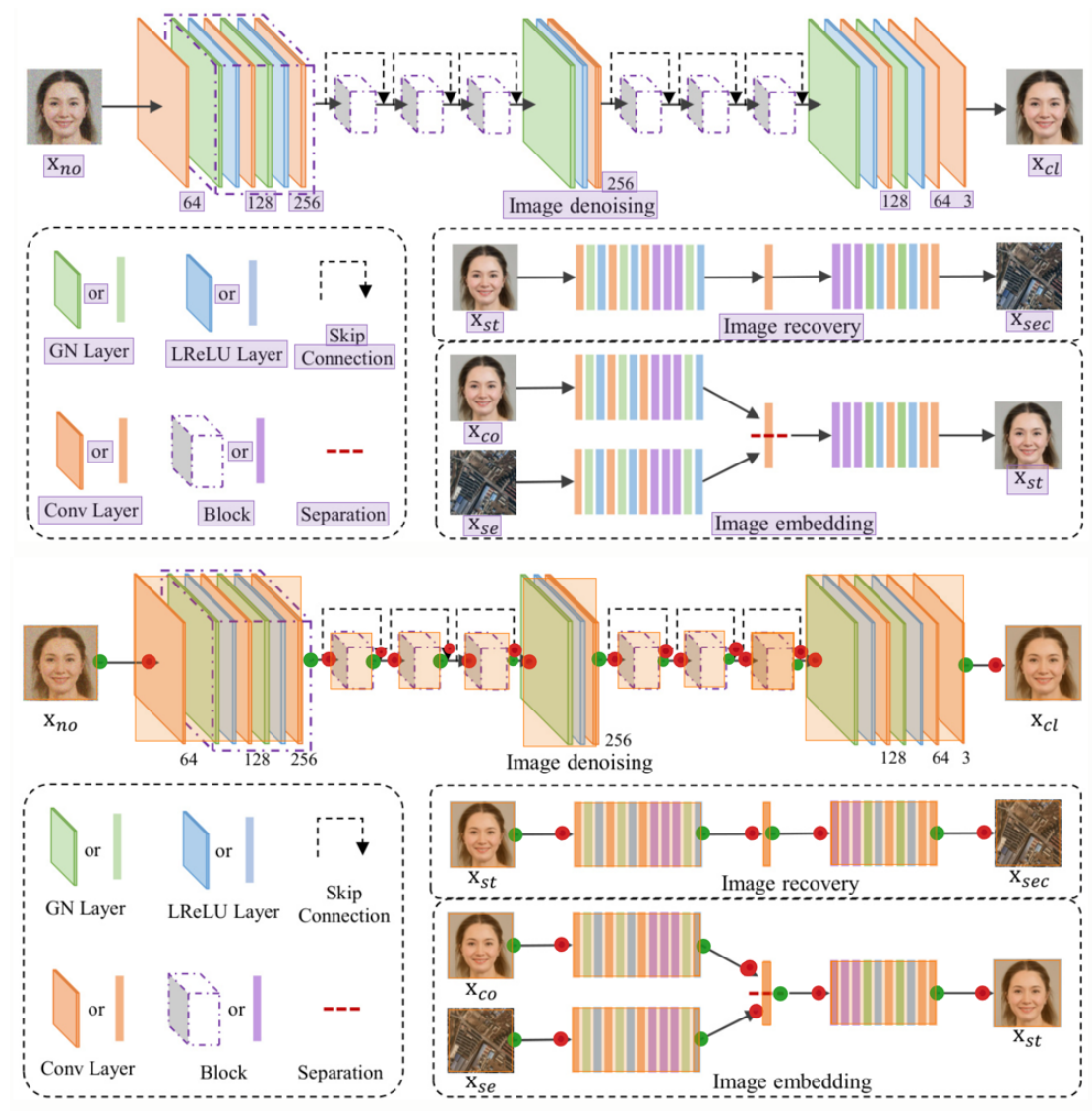


Figure 8: **Dataset annotation example 3.** Content and structural annotations for a real method figure.

Recognize the text inside each colored bounding box. Process boxes in this fixed color order:

red -> orange -> yellow -> green -> blue -> violet.

Return a JSON list of strings, with one string for each present bounding box. Skip missing colors and do not output empty strings for them. The list length K must equal the number of present colored boxes. The strings must follow the fixed color order above among the present boxes.

Text transcription rules:

- Preserve capitalization, punctuation, symbols, and spaces exactly as shown within each line.
- Replace line breaks within the same bounding box with a single space.
- Transcribe mathematical formulas as LaTeX code.
- Escape LaTeX backslashes as needed for valid JSON, e.g. `"\\textbackslash\\textbackslash frac\\{a\\}\\{b\\}"`.

Return only the required JSON list. Do not include explanations, markdown, or extra text.

Figure 9: **Prompt for Dense Content Perception.** The prompt asks the model to read highlighted text regions in a fixed color order and return the transcriptions as a JSON list.

The red bounding box and the blue bounding box each enclose exactly one node.

Is there a direct edge from the red-boxed node to the blue-boxed node?

Answer true if there is a direct edge whose direction allows movement from the red-boxed node to the blue-boxed node. This includes:

- a directed edge from red to blue;
- a direct edge with arrowheads at both ends, treated as bidirectional;
- a direct edge with no arrowhead, treated as bidirectional.

Answer false otherwise, including cases where the red-boxed node can reach the blue-boxed node only through intermediate nodes.

Return only true or false. Do not include explanations, markdown, or extra text.

Figure 10: **Prompt for Edge-Level Verification.** The prompt asks whether a directed edge exists from the red-boxed node to the blue-boxed node under the specified edge-direction rules.

The red bounding box encloses exactly one node.

How many direct edges should be counted as `\{incoming/outgoing\}` edges of the red-boxed node?

Counting rules:

- Count edges, not distinct neighboring nodes.
- Count duplicate or parallel edges separately.
- A directed edge counts as `\{incoming/outgoing\}` if its `\{arrowhead points to / tail is at\}` the red-boxed node.
- A direct edge with arrowheads at both ends is treated as bidirectional, so it counts as one `\{incoming/outgoing\}` edge.
- A direct edge with no arrowhead is treated as bidirectional, so it counts as one `\{incoming/outgoing\}` edge.
- A self-loop on the red-boxed node counts as one `\{incoming/outgoing\}` edge.

Return only the integer. Do not include explanations, markdown, or extra text.

Figure 11: **Prompt for Node-Level Degree Counting.** The prompt asks the model to count incoming or outgoing edges of the red-boxed node under the specified degree-counting rules.

The red bounding box and the blue bounding box each enclose exactly one node.

Is there a directed path from the red-boxed node to the blue-boxed node?

Path rules:

- A directed path is a sequence of nodes $v_0, v_1, \dots, v_n$, where v_0 is the red-boxed node, v_n is the blue-boxed node, and $n \geq 1$.
- The path may pass through zero or more intermediate nodes.
- Each edge has an allowed direction that depends on its arrowheads:
 - A directed edge may be followed only from its tail to its arrowhead.
 - An edge with arrowheads at both ends is treated as bidirectional and may be followed in either direction.
 - An edge with no arrowhead is treated as bidirectional and may be followed in either direction.
- For each consecutive pair v_i and v_{i+1} , there must be an edge directly connecting them whose allowed direction goes from v_i to v_{i+1} .
- A single edge whose allowed direction goes from the red-boxed node to the blue-boxed node counts as a valid path of length 1.

Return only true or false. Do not include explanations, markdown, or extra text.

Figure 12: **Prompt for Path-Level Reachability.** The prompt asks whether a directed path exists from the red-boxed node to the blue-boxed node under the specified path-following rules.

The node labeled " $\backslash\{\text{target}\}$ " is unique. Match " $\backslash\{\text{target}\}$ " exactly after trimming leading/trailing whitespace.

What are the texts of all direct neighbor nodes $\backslash\{\text{from which the node labeled "\{target\}" can be reached / that can be reached from the node labeled "\{target\}"\}$ by a single allowed edge? Return a JSON list of strings. Return [] if none.

Rules:

- Follow the allowed direction of each edge.
- A directed edge allows movement only from its tail to its arrowhead.
- An edge with arrowheads at both ends is treated as bidirectional.
- An edge with no arrowhead is treated as bidirectional.
- Only include direct neighbor nodes, not nodes reached through intermediate nodes.
- Return neighbor node texts, not edge counts.
- If different neighbor nodes have the same text, include each occurrence.
- If multiple parallel edges connect the same neighbor node and " $\backslash\{\text{target}\}$ ", include that neighbor text only once.
- Preserve capitalization, punctuation, symbols, and spaces exactly as shown; replace line breaks within one node label with a single space.
- Transcribe mathematical formulas as LaTeX code, escaping backslashes as needed for valid JSON, e.g. " $\backslash\text{textbackslash}\backslash\text{textbackslash}\frac{\{a\}}{\{b\}}$ ".
- The order of strings does not matter.

Return only the required JSON list. Do not include explanations, markdown, or extra text.

Figure 13: **Prompt for Information-Flow Tracing.** The prompt asks the model to list the text of direct incoming or outgoing neighbors for a uniquely named target node.

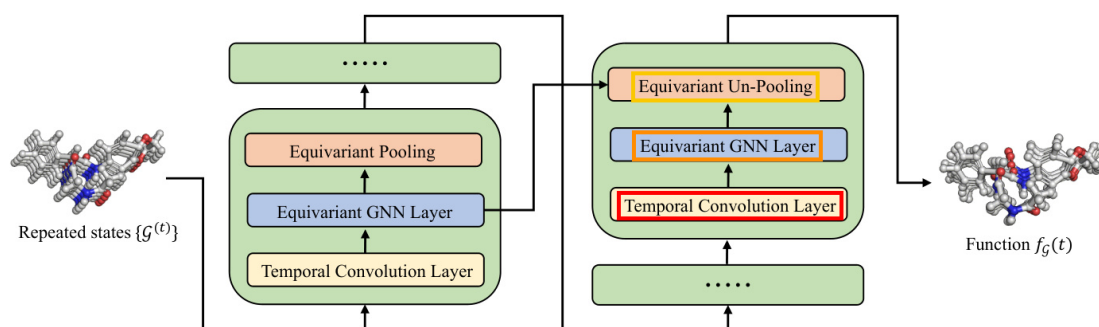


Figure 14: **Dense Content Perception failure for Qwen3.5-VL-27B.** The query asks the model to read the highlighted regions in the specified color order. The model correctly recognizes all three text contents but swaps the order of the latter two regions, returning an incorrectly ordered sequence. **Ground truth:** ["Temporal Convolution Layer", "Equivariant GNN Layer", "Equivariant Un-Pooling"]. **Model answer:** ["Temporal Convolution Layer", "Equivariant Un-Pooling", "Equivariant GNN Layer"].

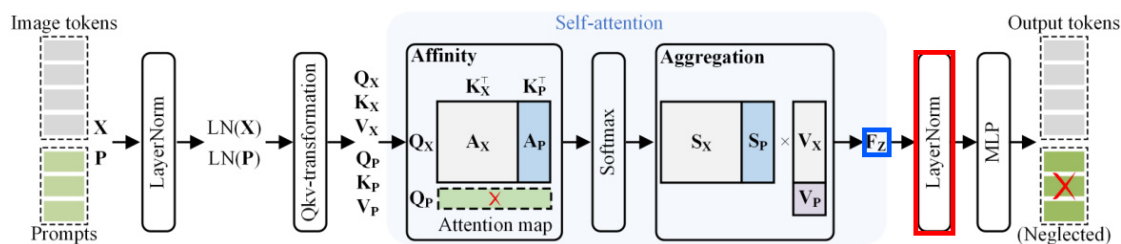


Figure 15: **Edge-Level Verification failure for InternVL3.5-8B.** The query asks whether a directed edge exists from the red-boxed node to the blue-boxed node. The figure instead contains the reverse blue-to-red connection, but the model answers "true" rather than the ground-truth "false", indicating a failure to correctly follow arrow direction.

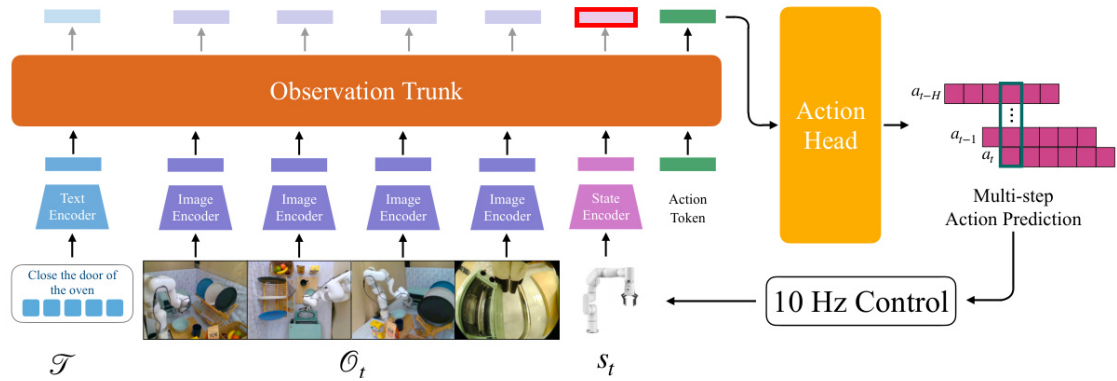


Figure 16: **Node-Level Degree Counting failure for Qwen3.5-VL-9B.** The query asks for the out-degree of the red-boxed node. The node has one incoming edge but no outgoing edge, yet the model answers 1 instead of the ground-truth 0, indicating confusion between incoming and outgoing connections.

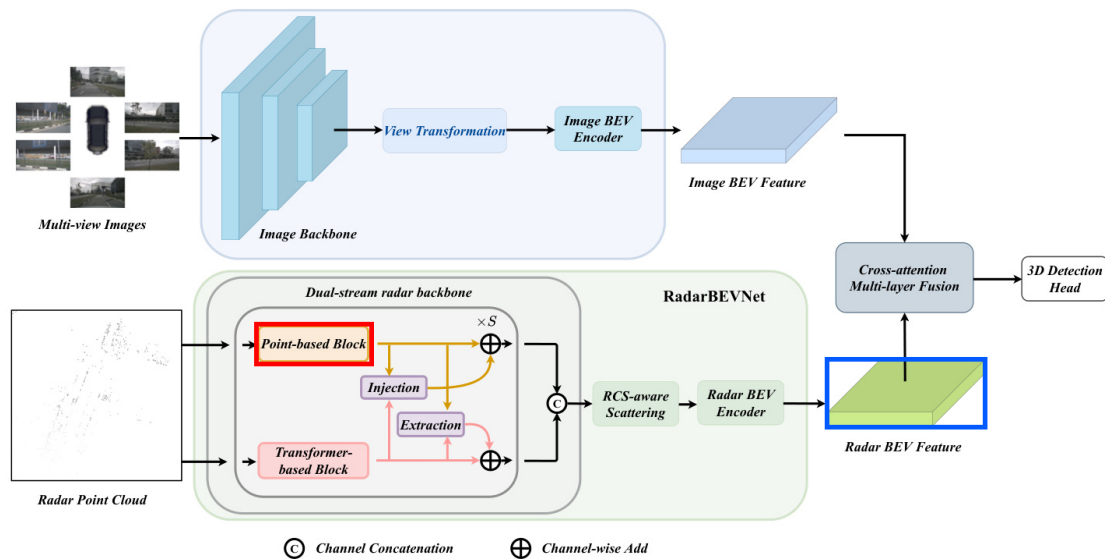


Figure 17: **Path-Level Reachability failure for Qwen3.5-VL-9B.** The query asks whether a directed path exists from the red-boxed node to the blue-boxed node. A multi-hop directed path connects the two nodes through intermediate components, yet the model answers "false" instead of the ground-truth "true", indicating a failure to compose multiple directed relations.

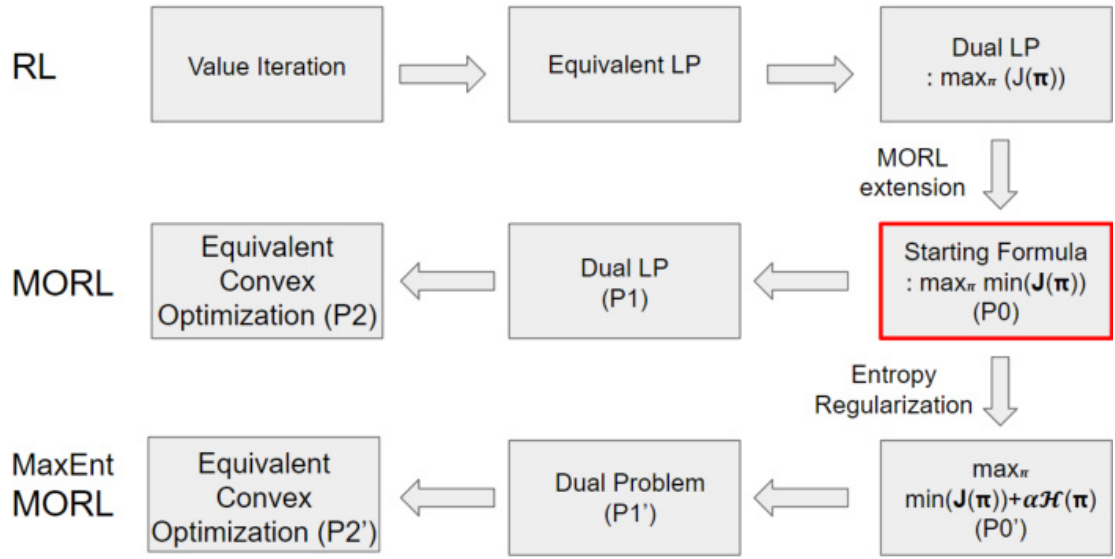


Figure 18: **Information-Flow Tracing failure for Qwen3.5-VL-9B.** The query asks for the direct neighbor reachable from the target node *Value Iteration* along the specified information-flow direction. The figure contains a direct edge from *Value Iteration* to *Equivalent LP*, yet the model returns an empty list instead of the ground-truth ["Equivalent LP"], indicating a failure to trace the outgoing information flow correctly.

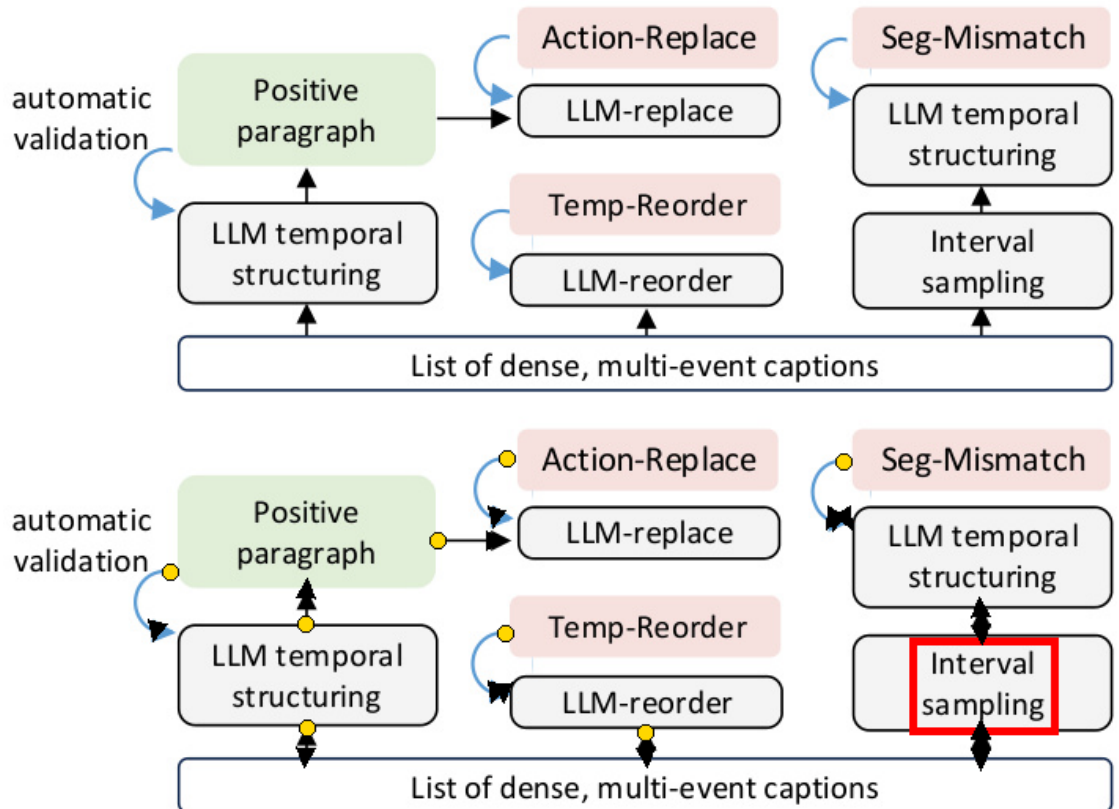


Figure 19: **Parser-induced cue error in Information-Flow Tracing for Qwen3.5-VL-9B.** The canonical input is shown above and the SciFigCue input is shown below. The model answers correctly under the canonical input, but fails under SciFigCue because incorrectly parsed arrow cues are overlaid near the selected node and obscure valid arrow evidence in the original figure. **Ground truth:** ["LLM temporal structuring"]. **Canonical answer:** ["LLM temporal structuring"]. **SciFigCue answer:** ["List of dense, multi-event captions"].

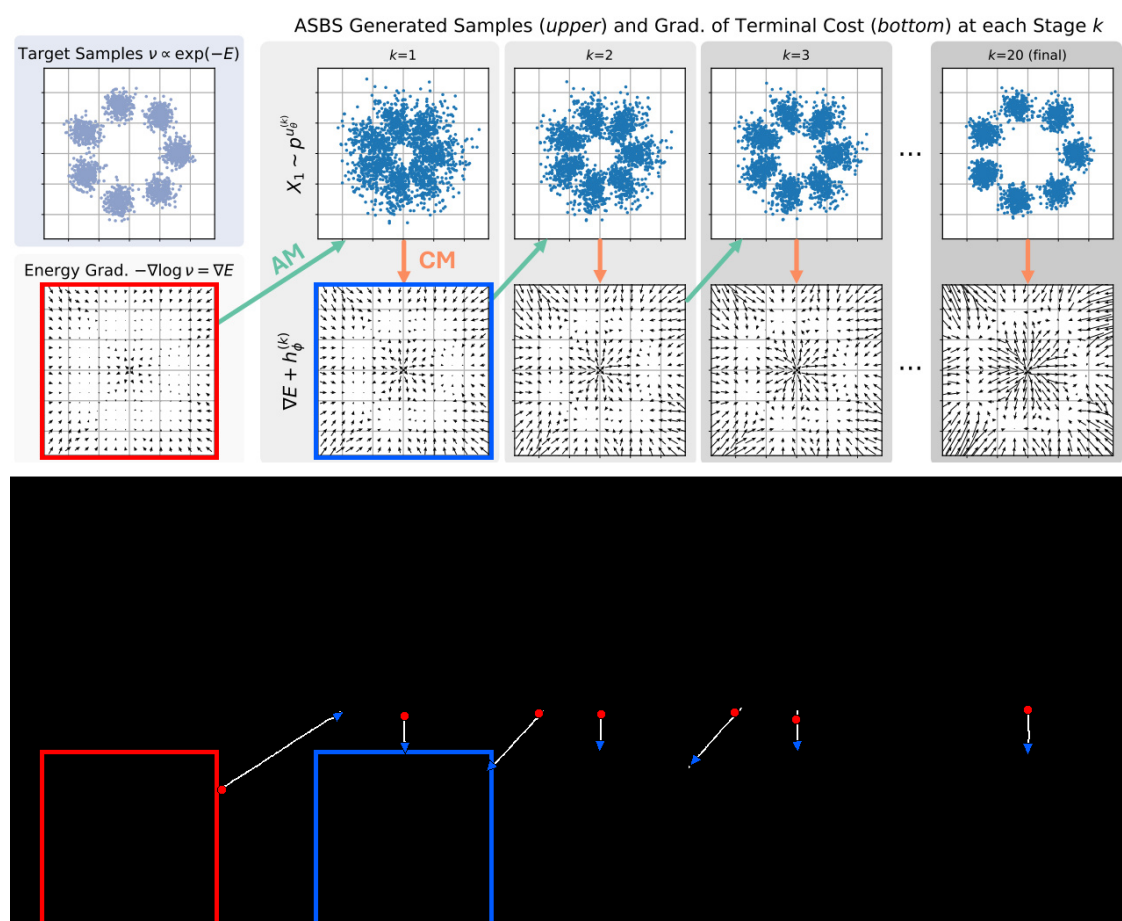


Figure 20: **Residual reasoning failure in Edge-Level Verification for Qwen3.5-VL-9B.** The canonical input is shown above, and SciFigCue augments it with the additional visual input shown below; both images are provided to the model. This example is a negative two-hop case asking whether a direct edge exists from the red-boxed node to the blue-boxed node. SciFigCue provides correct structural cues, but the two nodes are connected only through an intermediate node rather than by a direct edge. The model nevertheless answers "true" instead of the ground-truth "false", indicating persistent confusion between direct and indirect relations despite correct cues.