CAPE: A CLINICAL CHART AGENT FOR PRECISION VISION EXTRACTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Multimodal large language models (MLLMs) can answer questions about charts yet struggle to extract their quantitative content precisely and comprehensively. The difficulty is particularly pronounced in clinical charts, where specialized measurements appear within dense visual layouts. Accurate recovery of this information is essential for further quantitative analysis and evidence synthesis. To close this gap, we introduce CAPE, an MLLM-driven agent harness that combines semantic reasoning with pixel-level measurement for clinical chart extraction. The MLLM identifies the quantities to extract and locates the relevant chart elements. Dedicated vision tools follow this guidance to measure the plotted values and return reliability signals that inform remeasurement within a staged workflow. Record-level verification checks the completeness and organization of the extraction before the workflow concludes. To evaluate extraction quality, we develop a novel synthetic data generation approach that produces clinical charts paired with ground truth restricted to quantities readable or derivable from each figure. Using this approach, we construct a benchmark of 950 figures spanning 19 clinical chart types. CAPE raises the Overall extraction score on this benchmark from 0.68 to 0.78 over the strongest baseline built on the Claude Agent SDK and achieves higher scores on 17 of the 19 chart types, producing more complete and correctly organized records. CAPE also achieves the highest Overall score among all evaluated methods on a real clinical chart dataset.

1 INTRODUCTION

Clinical charts are a primary medium for communicating clinical and biomedical findings. For example, a Kaplan–Meier curve conveys survival outcomes, a forest plot summarizes treatment effects across patient subgroups, and dose-response curves characterize therapeutic efficacy. Beyond presenting study conclusions, these charts encode quantitative values and statistical measurements whose recovery supports re-analysis of published studies and evidence aggregation Guyot et al. (2012). However, this information is often accessible only through visual inspection, making accurate recovery a challenge for automated analysis. Existing tools, such as semi-automatic plot digitizers Rohatgi (2024), still require substantial manual intervention for each figure.

Clinical charts also present challenges that are not fully captured by existing chart-understanding benchmarks Masry et al. (2022); Methani et al. (2020). Two properties are particularly important. First, the quantities of interest are specialized and require precise extraction. Readers extract R^2, hazard ratios, confidence intervals, and Half Maximal Inhibitory Concentration (IC50) values, for which numerical errors can affect clinical interpretation. Second, the charts are visually dense. Survival curves carry step drops and censoring ticks, forest plots pack point estimates inside overlapping confidence intervals, and volcano plots scatter thousands of genes where only a few are significant. Such information is seldom transcribed in full elsewhere in the paper. Together, these characteristics demand both high numerical fidelity and the ability to reason over dense visual structures. Existing chart-to-table methods primarily address simpler charts such as bars and lines Liu et al. (2023a;b), leaving the density and precision of clinical figures insufficiently addressed.

Consequently, existing chart-reading methods address these demands only partially. Chart-specialized models are pretrained or fine-tuned on chart corpora Masry et al. (2023); Han et al. (2023), while recent agent systems add visual tools for finer perception Zhang et al. (2026). Al-

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.

though these approaches improve chart understanding, specialized clinical types and dense layouts remain challenging. General-purpose MLLMs Singh et al. (2025); Anthropic (2025); Comanici et al. (2025) interpret chart content but often read values coarsely, missing fluctuations along curves or the endpoints of crowded confidence intervals. Dedicated extraction pipelines use keypoint detection or segmentation to recover marks and lines more precisely Luo et al. (2021); Lal et al. (2023), but their coverage is restricted to particular chart types. Clinical chart extraction therefore requires both semantic interpretation and precise measurement within a method that accommodates diverse figures.

Evaluating clinical chart extraction is difficult because published figures rarely include their underlying values. These often originate from protected patient studies Kayaalp (2018), limiting access to ground truth for comparing extraction methods.

We therefore develop a reproducible synthetic generation approach that produces diverse clinical charts paired with ground truth restricted to readable or derivable quantities. We instantiate this approach in a 950-figure benchmark spanning 19 chart types, four difficulty tiers, and four rendering engines, evaluating structural completeness, textual labels, and numerical accuracy. Beyond evaluation, the approach can generate additional chart–data pairs for model training.

Building on this benchmark, we introduce **CAPE** (Clinical chart Agent for Precision Extraction) to address the complementary challenge of accurate extraction and cross-chart generalization. CAPE follows a staged workflow that classifies the chart, surveys its layout, extracts its contents, and verifies the resulting record (Figure 1). Within this workflow, the MLLM identifies the quantities to recover and directs precision vision tools to handle the relevant elements. The tools return measurements with reliability signals that guide further inspection and correction. Because perception is delegated to deterministic tools rather than learned separately for each chart type, CAPE supports extraction across diverse clinical figures.

CAPE sets a new state-of-the-art under identical scoring. On our benchmark it outperforms leading closed-source MLLMs and surpasses a strong agent baseline built on the Claude Agent SDK. It raises Overall extraction score from 0.68 to 0.78 over the strongest baseline and wins on 17 of the 19 chart types. The same advantage holds beyond synthetic data: CAPE also leads on a real and difficult clinical chart dataset.

The main contributions of our work are summarized as follows:

- **CAPE: an agent harness for precise and generalizable extraction.** We introduce CAPE, a staged extraction harness that combines MLLM reasoning with precision vision tools. By combining semantic interpretation with pixel-level measurement, CAPE supports fine-grained extraction across 19 clinical chart types.
- **Precision vision tools built for clinical charts.** We design 13 dedicated tools for the visual primitives these figures depend on. They trace curves, resolve error bars, and locate dense marker clouds. Each tool returns a reliability signal the agent can verify.
- **A reproducible clinical chart generation approach and benchmark.** We formalize the constituent elements of 19 clinical chart types in structured data models and generate diverse figures with ground truth restricted to readable or derivable quantities. The approach supports benchmarking and potential model training; we instantiate it in a 950-figure evaluation benchmark.
- **State-of-the-art performance on synthetic and real clinical charts.** CAPE outperforms the evaluated standalone MLLMs and the Claude Agent SDK baseline in Overall extraction score on both the synthetic benchmark and real clinical charts.

2 Related Work

2.1 Chart Understanding and Question Answering

Most chart research is framed as question answering. Early datasets built questions from templates over synthetic figures, such as FigureQA Kahou et al. (2017) and DVQA Kafle et al. (2018). Later benchmarks turned to real charts and open-ended questions. ChartQA Masry et al. (2022) and

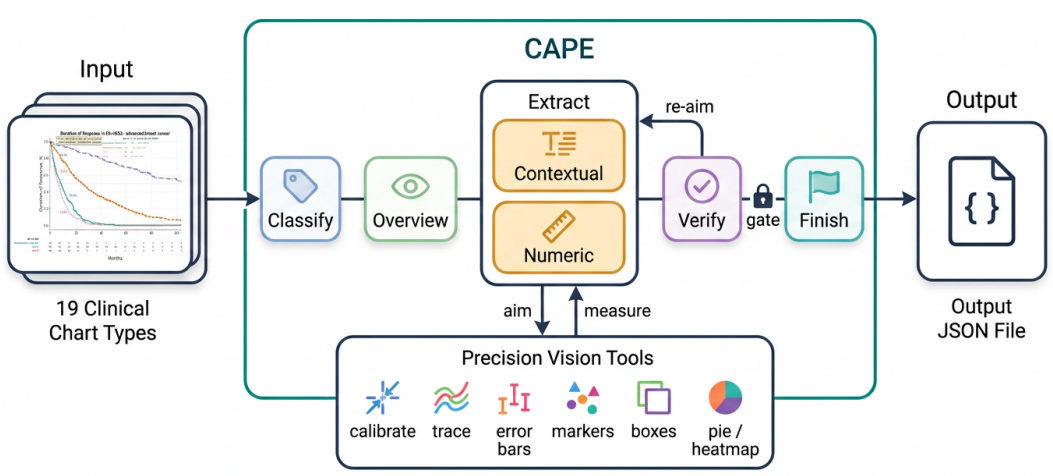


Figure 1: Overview of CAPE. An MLLM-driven agent harness processes a clinical chart through staged steps (Classify, Overview, Extract, Verify, Finish). The Extract step runs two parallel paths into one shared schema: contextual extraction reads printed text, while numeric extraction directs the precision vision tools to measure plotted values. Verify re-reads the chart and triggers re-aiming when a value is unreliable, and a completeness gate blocks finishing until every field is filled and correctly placed. The agent reasons and directs; deterministic vision tools measure.

PlotQA Methani et al. (2020) became the standard here. More recent efforts push the setting further with ChartX Xia et al. (2025), CharXiv Wang et al. (2024), and ChartQAPro Masry et al. (2025).

A correct final answer, however, does not reveal whether the model genuinely read the chart or simply inferred the answer from surrounding text. Recent studies bring this concern into focus. ChartR Chen et al. (2026) scores the full reasoning chain and identifies value extraction as the principal bottleneck. It further finds that models often lean on textual cues instead of the plotted values. Chart-FR1 Pan et al. (2026) studies dense, high-information charts and traces failure to weak fine-grained perception. START Liu et al. (2026) argues that spatial reading, not language alone, underpins true chart understanding. Our work departs from this line. Rather than answer a question, CAPE reconstructs the full underlying data. This goal hinges on exactly the precise perception these studies find lacking.

2.2 Chart Data Extraction and De-rendering

A separate line of work extracts structured content directly from the chart. Task-specific vision methods include ChartOCR Luo et al. (2021) for common chart types, LineFormer Lal et al. (2023) for instance-based line segmentation, and ChartZero Touhidul Islam et al. (2026) for line extraction trained on synthetic curves. Earlier work includes FigureSeer Siegel et al. (2016), which parses result figures in papers. Although these pipelines measure precisely, their coverage remains restricted to particular chart types. ChartArena Peng et al. (2026) further reports sharp accuracy drops under changes in style, format, or language.

A second branch de-renders the chart with a vision-language model. DePlot Liu et al. (2023a) and MatCha Liu et al. (2023b) translate plots into flat tables, with subsequent improvements from OneChart Chen et al. (2024) and TinyChart Zhang et al. (2024). ComProScanner Roy et al. (2026) extends this approach to scientific figures, while MM-ReCoder Tang et al. (2026) and PlotEdit Goswami et al. (2025) recover runnable plotting code. Neither output directly specifies the specialized fields required for clinical extraction. Flat tables lack an explicit clinical schema, whereas plotting code targets the figure's appearance. CAPE instead recovers clinically meaningful quantities within a typed record while retaining applicability across diverse chart types.

2.3 Tool-Augmented and Agentic Approaches

A growing line lets a model act with tools rather than answer in a single pass. Multimodal systems bring this idea to images. PixelCraft Zhang et al. (2026) inspects structured images with computer-vision routines and a fine-tuned grounding model, orchestrated by planner and critic agents. PlotEdit Goswami et al. (2025) edits a chart through several self-reflective agents that trade numeric, visual, and code feedback. Other methods guard against unreliable perception by aggregating repeated attempts. Self-Ensembling VLMs Berkane et al. (2026) sample a chart-to-table model many times and reconcile the outputs into a consensus.

These methods refine reasoning, chart reconstructions, or consensus tables through feedback. CAPE structures verification at two levels: tool-derived reliability signals guide correction of individual measurements, while a schema-based gate checks record completeness before completion.

3 Clinical Chart Benchmark

Evaluating clinical chart extraction requires ground-truth values that are rarely available from published figures. We therefore synthesize the data before rendering each chart, making its ground truth known by construction. This generation approach can produce additional chart–data pairs for evaluation and model training beyond the fixed benchmark studied here. Benchmark design follows three goals: correctness of the ground truth, breadth across clinical chart types, and visual realism close to published figures.

3.1 Data Model and Ground Truth

The benchmark spans the **19 chart types** that recur in clinical and biomedical publications, listed in Table 4 in Appendix A. For each type, we specify its constituent elements and extractable quantities in a structured data model, which guides a dedicated generator. For example, Kaplan–Meier generator draws survival curves from a declining at-risk population with censoring events; dose–response generator samples a sigmoidal curve with a plausible EC50; forest-plot generator produces per-subgroup effect sizes with confidence intervals. This grounding in a domain model is what makes a synthetic figure resemble a real one, rather than a random scatter of numbers.

The generator renders each figure from its data record, so every plotted element traces back to a known value. Generation is deterministic and reproducible, with the seed construction and regeneration details provided in Appendix A. In total the benchmark comprises **950 figures** across the 19 types and four difficulty tiers.

3.2 Rendering Variation

To discourage overfitting to a single visual style, we vary the plotting engine, difficulty, and layout used to render the same data.

Diversity. Each figure is drawn by one of four plotting engines: Matplotlib, Seaborn, Plotly, and plotnine. The engines differ in fonts, marker shapes, gridlines, and legend styling. Therefore, one data record can yield visibly different figures. Their coverage is summarized in Table 5 in Appendix A. Each figure is rendered at random by one of the engines that support its chart type. Appearance varies further through randomized style attributes, including figure size, color palette, fonts, marker sizes, and resolution.

Difficulty tiers. Drawing on prior characterizations of chart visual complexity (Yang et al., 2025; Kaur et al., 2025), we define four difficulty tiers through rendering presets that vary chart density, typography, annotations, and axis properties (Table 6). The tiers range from clean, well-spaced charts to dense, heavily annotated figures, testing both typical performance and fine-grained reading at the limits of legibility. Each setting stays within what a chart type can naturally express, keeping even the hardest figures realistic.

Poster-style realism. Real clinical figures rarely appear as a bare plot. They carry the contextual furniture of a paper or a conference poster, including a title placed above the figure, a legend outside

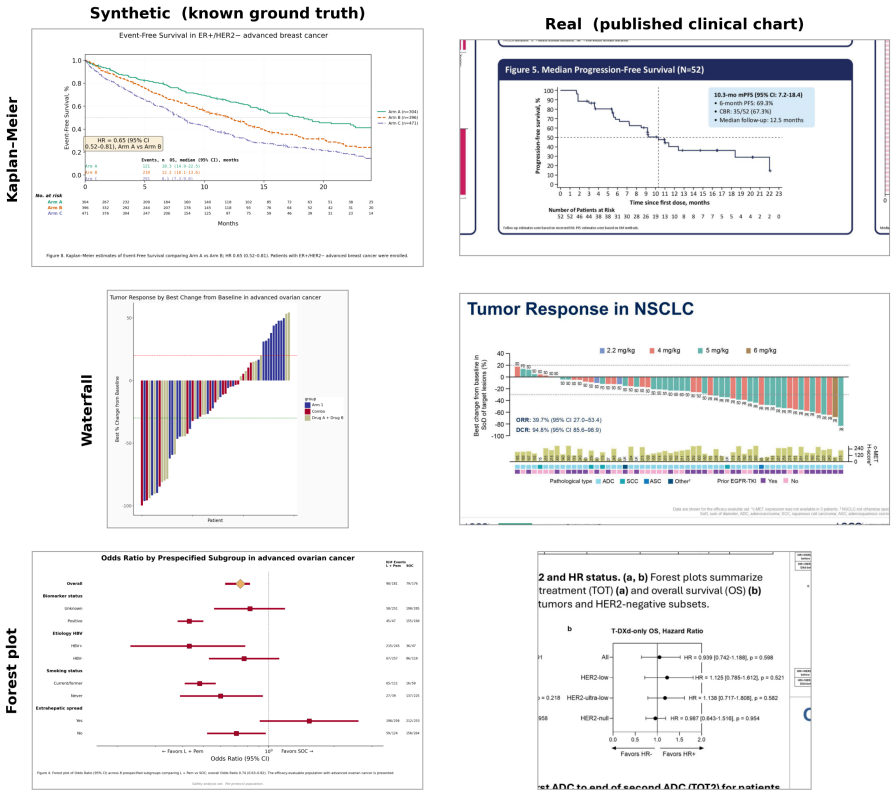


Figure 2: Synthetic benchmark figures (left of each pair) beside real published charts (right) for Kaplan–Meier, waterfall, and forest plots. The synthetic figures follow the visual conventions of published clinical charts while carrying exact ground truth.

the axes, a caption, footnotes, an abbreviation key, and in-plot annotations such as a hazard-ratio box. We reproduce this furniture on top of each chart and place it with a collision-aware pass, so that annotations settle into free space rather than over the data. Each element appears with a probability that rises with the difficulty tier. The result closes much of the visual gap to published figures. Figure 2 places synthetic figures beside real published charts of the same type.

3.3 Readable and Derivable Ground Truth

A generator knows every latent parameter behind a figure, but a reader does not. The ground truth is therefore restricted to what the figure can actually convey. This covers two kinds of quantities. The first is shown directly, such as a plotted value, an axis label, or a text annotation. The second is not printed but can be derived from what is shown, such as a median survival read from where a Kaplan–Meier curve crosses one half. Recovering the second kind demands genuine understanding of the chart rather than mere transcription, which is precisely the capability we aim to measure.

4 CAPE

CAPE turns a clinical chart into a typed data record by combining MLLM reasoning with precision vision tools. Within a staged workflow (Section 4.1), the agent directs pixel-level measurement (Section 4.2) and verifies the resulting record for completeness and consistency (Section 4.3).

4.1 Overview and Workflow

CAPE processes a figure through five stages, shown in Figure 1. Appendix Figure 3 illustrates these stages through a worked Kaplan–Meier extraction, including tool feedback and subsequent refinement.

Classify. Each run opens by selecting one of the 19 clinical chart types, or a generic family when none applies. To guard against a near miss, the agent is shown the distinguishing features of similar types before it commits. Selecting a type loads the schema and fixes the fields the extraction must fill. Classification therefore decides what the record should contain, not how to read the figure. From there, the agent plans the reading itself without hardcoded rules specifying which element to read or where to aim a tool. Because the same staged workflow runs on every type, a chart outside the 19 is read no differently.

Overview. Before any value is read, the agent surveys the figure as a whole. It records the distinct regions or panels, the role played by each legend entry, and whether the panels report different endpoints or merely different views of the same subjects. Guided by this survey, the agent organizes the schema and reads each region separately so that its values settle into the appropriate part of the record.

Extract. Once the schema is organized, the agent fills it by reading titles, axis labels, legends, units, and any numerical values explicitly stated in the figure. An annotated value is recorded directly without calling a vision tool to measure the corresponding plotted element. When a required value is not stated in text, the agent locates the corresponding curve, marker, or other plotted element. It then directs a vision tool to measure the value for the appropriate schema field. Each tool returns its measurement together with reliability signals that support verification and guide remeasurement when needed (Section 4.2).

Verify and Finish. Once the schema is filled, the agent revisits the overview to confirm that every region and panel the survey identified is accounted for. Any record that still leaves a field empty or a value out of place is held back, requiring the agent to recover what it missed (Section 4.3). Only a complete record is emitted as output.

4.2 Precise Extraction with Vision Tools

Reading each value precisely is where prior methods fall short. A multimodal model understands a figure but reads it imprecisely. A vision pipeline measures precisely but cannot understand the figure. CAPE combines both strengths through a strict protocol between the agent and the tools.

The aim-and-measure protocol. The agent identifies the relevant chart element, marks its approximate location, and specifies the target schema field. A vision tool refines this location and measures the value against calibrated axes. This separates semantic guidance from precise pixel localization. In Appendix Figure 3, for instance, the tool snaps an agent-supplied rough polyline to nearby Kaplan–Meier curve pixels. Shared geometric primitives allow the same tools to serve different chart types: one tracing routine handles Kaplan–Meier, dose–response, and pharmacokinetic curves. The agent selects the appropriate measurements without a separate reading procedure for each type.

Measurements that carry evidence. Our tools return their results with reliability signals derived from the measurement itself. A curve trace, for example, reports its coverage and flags abrupt jumps that may indicate a crossing onto a neighboring curve. When a signal points to an unreliable measurement, the agent re-aims the tool or measures anew before committing the value. Appendix B describes the tool-specific signals, with the full inventory in Table 7.

Failure modes addressed. Reliable measurement begins with axis calibration. Earlier extraction pipelines match detected tick positions to axis values (Luo et al., 2021; Rohatgi, 2024). CAPE instead fits the pixel-to-value mapping from value-location anchors supplied by the agent, using

logarithmic coordinates for log axes. Local refinement improves anchor placement, while fit residuals and per-anchor confirmation support verification of the ruler. Beyond calibration, the tools separate dense marker clouds, follow overlapping same-color lines, and sample colors from agent-marked points rather than color names. CAPE uses these operations to measure chart elements that are difficult to distinguish through visual interpretation alone. Appendix B.2 provides the detailed procedures and safeguards for all four cases.

4.3 Robust and Complete Extraction

A precise reading of each value is necessary but not sufficient. A run must also terminate with a record that is complete and correctly organized. CAPE secures this through staged control and two distinct forms of verification.

Staged control. The staged workflow of Section 4.1 is realized as a state machine in the spirit of StateFlow (Wu et al., 2024). This staging is what makes the extraction robust. Each state passes a settled result to the next. Its quality therefore carries forward rather than being reworked downstream. The same principle guards the end of the run. A record is emitted only after it passes verification. Should it fall short, the run returns to extraction to supply what is missing. Extraction concludes only on a result that meets the standard. CAPE uses feedback for remeasurement within a fixed workflow; policy learning remains future work (Appendix B.4).

Two forms of verification. Verification operates at two levels. The first concerns a single measurement, as described in Section 4.2. Every tool reports evidence. The agent re-aims or re-measures a reading the evidence marks unreliable. This secures the accuracy of one value. The second concerns the record as a whole and draws on the idea of Reflexion (Shinn et al., 2023), in which an agent reflects on its own output before acting on it. Once the schema is filled, the agent revisits its result in this way before the run may finish. A deterministic gate first checks the record for completeness without consulting the image, flagging any field left empty or placed in the wrong slot. The agent then re-reads the figure to confirm what the gate cannot see, such as whether the item count matches the chart and whether each region's data sits where it belongs. These steps let the agent catch and repair its own errors before they reach the output.

Verification also uses the provenance stored with each value to trace its source and distinguish a deliberate empty field from one that was never measured. Landmark checks test values against constraints visible in the chart, while round-trip checks project extracted values back onto their source marks. These checks supplement the record-level review without requiring ground truth. Appendix B.3 describes the provenance records and both checks in detail.

5 Experiments

5.1 Evaluation Metric

We evaluate each extracted record along three axes. Structure measures the fraction of ground-truth top-level fields present in the prediction. Labels measures the mean token-level F1 of textual fields (Rajpurkar et al., 2016). Numeric scores values by relative error, with fields weighted by the amount of data they contain. Array elements are matched before scoring, with missing and extra elements penalized. The three axes combine into an Overall score,

$$\text{overall} = \lambda_1 \,\text{numeric} + \lambda_2 \,\text{labels} + \lambda_3 \,\text{structure}, \qquad \lambda_1 + \lambda_2 + \lambda_3 = 1. \tag{1}$$

We set $\lambda_1 = 0.4$ and $\lambda_2 = \lambda_3 = 0.3$, giving numerical accuracy the largest weight. Field names are reconciled before scoring, and generator-only fields that cannot be recovered from the figure are excluded. Appendix C provides the full scoring definitions and matching procedure.

5.2 Setup and Baselines

All methods follow the same protocol: given a figure, a method classifies its chart type and then extracts the underlying data into the target schema. We compare our method against two kinds of baseline that represent how such extraction is commonly attempted today. The first baseline

family uses a general MLLM directly. The model is shown the figure together with the target schema and returns the filled schema in a single response, without external tools. We evaluate three models of comparable capability: GPT-5.5 (`gpt-5-5-2026-04-24`), Gemini 3 Flash (`gemini-3-flash-preview`), and Claude Opus 4.8 (`claude-opus-4-8`). The second baseline is a general-purpose coding agent, Claude Code, built on the Claude Agent SDK and driven by Claude Opus 4.8. Unlike the single-pass models, it can view the figure and write and run code over several turns before producing the schema, though it is not specialized for chart reading. Our method, CAPE, uses the same Claude Opus 4.8 backbone as the coding agent but couples the model with dedicated vision tools in a staged extraction pipeline. Both agents are given an iteration budget large enough to finish the task; the comparison therefore reflects extraction ability rather than a step limit.

Table 1: Extraction quality on the 950-figure synthetic benchmark. Best per column in bold.

Method	Structure	Labels	Numeric	Overall
GPT-5.5	0.700	0.760	0.599	0.678
Gemini 3 Flash	0.558	0.605	0.502	0.550
Claude Opus 4.8	0.657	0.728	0.541	0.632
Agent SDK (Claude Opus 4.8)	0.650	0.727	**0.667**	0.680
CAPE, ours (Claude Opus 4.8)	**0.867**	**0.851**	0.662	**0.780**

5.3 RESULTS

Table 1 and Table 2 report extraction quality on the synthetic benchmark and on real clinical charts. In both settings CAPE achieves the highest Overall score. On the synthetic benchmark, CAPE improves Structure and Labels while maintaining numerical accuracy comparable to the strongest baseline. On real clinical charts, it leads Numeric and Labels, with all methods tied on Structure.

The two baseline families reveal where the difficulty lies. The single-pass MLLMs read the figure in one forward pass and achieve Numeric scores of 0.50 to 0.60 on the synthetic benchmark, indicating that direct transcription remains insufficient for precise numerical extraction. The Claude Agent SDK baseline closes part of this gap: with the freedom to inspect the figure and run code over multiple turns it reaches the best Numeric score among the baselines (0.667), which shows that tool use rather than model scale is what unlocks accurate measurement. Yet that same agent remains limited on Structure and Labels. Receiving the target schema alone does not ensure that the extracted values are complete and correctly organized. CAPE addresses these limitations through its staged workflow and record-level verification. Its staged pipeline resolves the schema layout and the axis metadata before reading, lifting Structure to 0.867 and Labels to 0.851 far above every baseline, while its dedicated vision tools maintain comparable Numeric performance to the strongest agent. Combining accurate values with a correctly assembled record raises Overall to 0.780, roughly ten points ahead of the best baseline.

Table 8 in Appendix D compares CAPE with the Claude Agent SDK baseline across all 19 chart types. CAPE achieves a higher Overall score on 17 types. The largest gains occur on Multi-panel ($+0.380$), Spider web ($+0.240$), and PK curves ($+0.205$), where extraction requires resolving complex layouts or distinguishing multiple series.

Table 2: Extraction performance on real clinical charts depicting adverse event severity. Best per column in bold.

Method	Structure	Labels	Numeric	Overall
GPT-5.5	**1.000**	0.367	0.193	0.487
Gemini 3 Flash	**1.000**	0.367	0.126	0.460
Claude Opus 4.8	**1.000**	0.343	0.206	0.485
Agent SDK (Claude Opus 4.8)	**1.000**	0.390	0.143	0.474
CAPE, ours (Claude Opus 4.8)	**1.000**	**0.400**	**0.219**	**0.508**

CAPE retains the highest Overall score on the real clinical figures. Table 2 reports the results. CAPE stays first on every scored axis. It leads Overall (0.508), Numeric (0.219), and Labels (0.400),

while matching the top Structure score (1.000). GPT-5.5 follows at 0.487 and Claude Opus 4.8 at 0.485. The Claude Agent SDK baseline reaches 0.474. Numerical extraction remains difficult across methods on these dense figures, which contain up to nine small-multiple panels within an image only a few hundred pixels tall. Every panel stacks three to four overlapping severity curves. One figure can hold as many as twenty-seven. These curves are thin, jagged weekly steps drawn across a year of follow-up. Sparse ticks and blurred lines within these small panels make precise reading difficult. Despite these challenges, CAPE achieves the highest Numeric, Labels, and Overall scores.

Because every method reaches Structure 1.000, differences in Overall on these figures arise from Numeric and Labels. Comparing the systems with a shared backbone helps examine the differences in numerical extraction. Three of the systems share the same Claude Opus 4.8 backbone. Any difference in their reading therefore reflects how each is allowed to look, not model scale. Read in a single forward pass, the plain model scores 0.206 on Numeric, recovering the shape of a curve but not its value week by week. Given general-purpose tools and sixteen to thirty steps per figure, the Claude Agent SDK baseline can write code to inspect pixels. Yet without a measurement primitive tied to the axes, it drifts to 0.143, below even the single-pass model. CAPE instead fits each panel's axes to within a few pixels and hands every curve to a dedicated tracer, reading its values off the plot rather than inferring them from appearance. This lifts Numeric to 0.219, the best of the field. The pattern shows that the advantage comes from grounded measurement rather than from agency alone.

Table 3: Component ablations on the 76 charts evaluation set, ordered by Overall score. A checkmark indicates an enabled component and a cross a disabled component. Meas. sig. denotes measurement reliability signals. All configurations retain the MLLM controller.

Enabled components					Extraction scores	
Overview	Staged control	Reflexion	Meas. sig.	CV tools	Numeric	Overall
✓	✓	✓	✓	×	0.639	0.770
✓	×	✓	✓	✓	0.661	0.774
×	×	×	✓	✓	0.655	0.775
✓	✓	✓	✓	✓	**0.675**	**0.784**

5.4 Ablation Study

Table 3 examines the contribution of CAPE's components on a small evaluation set generated using the benchmark code. The set contains 76 charts, with four figures per chart type and one at each difficulty tier. The largest loss in numerical accuracy comes from removing the CV tools. Numeric falls from 0.675 to 0.639, with Overall dropping from 0.784 to 0.770. Even with the staged workflow and verification in place, the agent lacks the pixel-level measurements needed for precise reading. Keeping the tools while removing staged control yields Numeric 0.661 and Overall 0.774, both below full CAPE. Staged control thus helps the agent use the tools throughout extraction. With Overview, staged control, and Reflexion all removed, the configuration retaining CV tools and measurement signals reaches Numeric 0.655 and Overall 0.775. Restoring the complete workflow raises these scores to 0.675 and 0.784. CAPE achieves its best extraction performance when precise measurement is guided by the complete staged workflow.

6 Conclusion

CAPE combines MLLM interpretation with precise vision tools for clinical chart extraction. The agent directs measurements and uses tool feedback within a staged workflow to verify values and record completeness. On our benchmark of 950 figures across 19 clinical chart types, CAPE outperforms leading closed-source MLLMs and a strong agent baseline in Overall score, with its largest gains over the agent baseline in Structure and Labels and comparable numerical accuracy. CAPE also achieves the highest Overall score on real clinical figures. Shared vision tools support extraction across chart types without separate learned perception modules. CAPE supports recovery of analyzable data from published figures, while our reproducible chart generator supports benchmarking and future model training.

AI USE STATEMENT

We used generative AI to polish the manuscript, assist in preparing tables, equations, and algorithm descriptions, and identify potentially relevant related work. The authors reviewed and verified all AI-assisted content and take responsibility for the final manuscript.

REPRODUCIBILITY STATEMENT

The benchmark generation procedure, CAPE workflow, and evaluation protocol are described in Sections 3, 4, and 5, respectively. The code for the benchmark and CAPE will be made available upon publication of the paper.

REFERENCES

Anthropic. Csystem card: Claude opus 4 & claude sonnet 4. `https://www.anthropic.com/claude-4-system-card`, 2025.

Thomas Berkane, Qianyi Wang, and Maimuna S Majumder. Self-ensembling vision-language models for chart data extraction. *arXiv preprint arXiv:2605.27298*, 2026.

Jinyue Chen, Lingyu Kong, Haoran Wei, Chenglong Liu, Zheng Ge, Liang Zhao, Jianjian Sun, Chunrui Han, and Xiangyu Zhang. Onechart: Purify the chart structural extraction via one auxiliary token. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pp. 147–155, 2024.

Xiaojun Chen, Sixiao Luo, Ziqi Liu, Min Yang, Qin Zhang, and Liang-Jie Zhang. Chartr: Evaluating reasoning accuracy and robustness in chart question answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 41193–41202, 2026.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

Kanika Goswami, Puneet Mathur, Ryan Rossi, and Franck Dernoncourt. Plotedit: Natural language-driven accessible chart editing in pdfs via multimodal llm agents. In *European Conference on Information Retrieval*, pp. 130–134. Springer, 2025.

Patricia Guyot, A. E. Ades, Mario J. N. M. Ouwens, and Nicky J. Welton. Enhanced secondary analysis of survival data: reconstructing the data from published Kaplan-Meier survival curves. *BMC Medical Research Methodology*, 12(1):9, 2012. doi: 10.1186/1471-2288-12-9.

Yucheng Han, Chi Zhang, Xin Chen, Xu Yang, Zhibin Wang, Gang Yu, Bin Fu, and Hanwang Zhang. Chartllama: A multimodal llm for chart understanding and generation. doi: 10.48550. *arXiv preprint arXiv.2311.16483*, 2023.

Kushal Kafle, Brian Price, Scott Cohen, and Christopher Kanan. Dvqa: Understanding data visualizations via question answering. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5648–5656. IEEE, 2018.

Samira Ebrahimi Kahou, Vincent Michalski, Adam Atkinson, Ákos Kádár, Adam Trischler, and Yoshua Bengio. Figureqa: An annotated figure dataset for visual reasoning. *arXiv preprint arXiv:1710.07300*, 2017.

Rachneet Kaur, Nishan Srishankar, Zhen Zeng, Sumitra Ganesh, and Manuela Veloso. Chartagent: A multimodal agent for visually grounded reasoning in complex chart question answering. *arXiv preprint arXiv:2510.04514*, 2025.

Mehmet Kayaalp. Patient privacy in the era of big data. *Balkan medical journal*, 35(1):8–17, 2018.

Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

Jay Lal, Aditya Mitkari, Mahesh Bhosale, and David Doermann. Lineformer: Line chart data extraction using instance segmentation. In *International conference on document analysis and recognition*, pp. 387–400. Springer, 2023.

Fangyu Liu, Julian Eisenschlos, Francesco Piccinno, Syrine Krichene, Chenxi Pang, Kenton Lee, Mandar Joshi, Wenhu Chen, Nigel Collier, and Yasemin Altun. Deplot: One-shot visual language reasoning by plot-to-table translation. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 10381–10399, 2023a.

Fangyu Liu, Francesco Piccinno, Syrine Krichene, Chenxi Pang, Kenton Lee, Mandar Joshi, Yasemin Altun, Nigel Collier, and Julian Eisenschlos. Matcha: Enhancing visual language pretraining with math reasoning and chart derendering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12756–12770, 2023b.

Zhuoming Liu, Xiaofeng Gao, Feiyang Niu, Qiaozi Gao, Liu Liu, and Robinson Piramuthu. Start: Spatial and textual learning for chart understanding. In *2026 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 8146–8156. IEEE, 2026.

Junyu Luo, Zekun Li, Jinpeng Wang, and Chin-Yew Lin. Chartocr: Data extraction from charts images via a deep hybrid framework. In *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1916–1924. IEEE, 2021.

Ahmed Masry, Jia Qing Tan, Shafiq Joty, Enamul Hoque, et al. Chartqa: A benchmark for question answering about charts with visual and logical reasoning. In *Findings of the association for computational linguistics: ACL 2022*, pp. 2263–2279, 2022.

Ahmed Masry, Parsa Kavehzadeh, Enamul Hoque, Shafiq Joty, et al. Unichart: A universal vision-language pretrained model for chart comprehension and reasoning. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pp. 14662–14684, 2023.

Ahmed Masry, Mohammed Saidul Islam, Mahir Ahmed, Aayush Bajaj, Firoz Kabir, Aaryaman Kartha, Md Tahmid Rahman Laskar, Mizanur Rahman, Shadikur Rahman, Mehrad Shahmohammadi, et al. Chartqapro: A more diverse and challenging benchmark for chart question answering. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 19123–19151, 2025.

Nitesh Methani, Pritha Ganguly, Mitesh M Khapra, and Pratyush Kumar. Plotqa: Reasoning over scientific plots. In *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1516–1525. IEEE, 2020.

Hongkun Pan, Yuwei Wu, Wanyi Hong, Shenghui Hu, Qitong Yan, Yi Yang, Rufei Han, Changju Zhou, Minfeng Zhu, Dongming Han, et al. Chart-fr1: Visual focus-driven fine-grained reasoning on dense charts. *arXiv preprint arXiv:2605.01882*, 2026.

Shangpin Peng, Gengluo Li, Xingyu Wan, Chengquan Zhang, Hao Feng, Binghong Wu, Huawen Shen, Weinong Wang, Ziyi Cai, Zhuotao Tian, et al. Chartarena: Benchmarking chart parsing across languages, scenarios, and formats. *arXiv preprint arXiv:2606.01348*, 2026.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pp. 2383–2392, 2016.

Ankit Rohatgi. Webplotdigitizer, 2024. URL `https://automeris.io`.

Aritra Roy, Enrico Grisan, Chiara Gattinoni, and John Buckeridge. Beyond text and tables: Vision-language model integration in comproscanner for extracting materials data from scientific figures with high accuracy. *arXiv preprint arXiv:2606.00065*, 2026.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in neural information processing systems*, 36:8634–8652, 2023.

Noah Siegel, Zachary Horvitz, Roie Levin, Santosh Divvala, and Ali Farhadi. Figureseer: Parsing result-figures in research papers. In *European Conference on Computer Vision*, pp. 664–680. Springer, 2016.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, et al. Openai gpt-5 system card. *arXiv preprint arXiv:2601.03267*, 2025.

Zitian Tang, Xu Zhang, Jianbo Yuan, Yang Zou, Varad Gunjal, Songyao Jiang, and Davide Modolo. Mm-recoder: Advancing chart-to-code generation with reinforcement learning and self-correction. *arXiv preprint arXiv:2604.01600*, 2026.

Md Touhidul Islam, Yasir Mahmud, Sujan Kumar Saha, Mark Tehranipoor, and Farimah Farahmandi. Chartzero: Synthetic priors enable zero shot chart data extraction. *arXiv e-prints*, pp. arXiv–2605, 2026.

Zirui Wang, Mengzhou Xia, Luxi He, Howard Chen, Yitao Liu, Richard Zhu, Kaiqu Liang, Xindi Wu, Haotian Liu, Sadhika Malladi, et al. Charxiv: Charting gaps in realistic chart understanding in multimodal llms. *Advances in Neural Information Processing Systems*, 37:113569–113697, 2024.

Yiran Wu, Tianwei Yue, Shaokun Zhang, Chi Wang, and Qingyun Wu. Stateflow: Enhancing llm task-solving through state-driven workflows. *arXiv preprint arXiv:2403.11322*, 2024.

Renqiu Xia, Hancheng Ye, Xiangchao Yan, Qi Liu, Hongbin Zhou, Zijun Chen, Botian Shi, Junchi Yan, and Bo Zhang. Chartx & chartvlm: A versatile benchmark and foundation model for complicated chart reasoning. *IEEE Transactions on Image Processing*, 2025.

Cheng Yang, Chufan Shi, Yaxin Liu, Bo Shui, Junjie Wang, Mohan Jing, Linran Xu, Xinyu Zhu, Siheng Li, Yuxiang Zhang, et al. Chartmimic: Evaluating lmm's cross-modal reasoning capability via chart-to-code generation. In *International Conference on Learning Representations*, volume 2025, pp. 26590–26646, 2025.

Liang Zhang, Anwen Hu, Haiyang Xu, Ming Yan, Yichen Xu, Qin Jin, Ji Zhang, and Fei Huang. Tinychart: Efficient chart understanding with visual token merging and program-of-thoughts learning. *arXiv preprint arXiv:2404.16635*, 2024.

Shuoshuo Zhang, Zijian Li, Yizhen Zhang, Jingjing Fu, Lei Song, Jiang Bian, Jun Zhang, Yujiu Yang, and Rui Wang. Pixelcraft: A multi-agent system for high-fidelity visual reasoning on structured images. In *International Conference on Learning Representations*, volume 2026, pp. 77608–77632, 2026.

A Clinical Chart Benchmark Details

The following details complement Section 3 by specifying deterministic generation, clinical chart coverage, and rendering variation. Together they describe how the benchmark preserves known ground truth while varying the appearance and complexity of the figures.

Deterministic generation. Each figure draws from a single random stream seeded by a pure function of its chart type, difficulty tier, and replicate index,

$$\text{seed} \;=\; \big(\text{base} + 7919\, i_{\text{type}} + 1000\, i_{\text{tier}} + r\big) \bmod 2^{31}, \tag{2}$$

where i_{type} and i_{tier} index the chart type and tier and r is the replicate. The same command reproduces byte-identical figures and ground truth. A manifest records all information needed to regenerate the benchmark.

Clinical chart coverage. The benchmark covers 19 chart types used to present clinical and biomedical findings. Table 4 summarizes their clinical roles, from survival outcomes and treatment effects to patient trajectories and study disposition. Each type is paired with the data model and ground truth described in Section 3.1.

Table 4: The 19 clinical chart types in the benchmark and the clinical role each serves.

Chart type	Clinical role
Kaplan–Meier	Survival over time by treatment arm, with censoring
Forest plot	Effect size with confidence interval across subgroups
Waterfall	Best per-patient tumor change from baseline
Swimmer	Per-patient treatment timeline and response
Oncology spider	Per-patient tumor-change trajectories over time
PK curves	Drug concentration over time by dose
Dose–response	Effect versus dose (sigmoidal, EC50/IC50)
Exposure–response	Response versus drug exposure per patient
Volcano	Differential expression: effect size versus significance
Heatmap	Expression or correlation across two categorical axes
Longitudinal	Repeated measures over time by arm
Line plot	Trend of a measure over time
Bar plot	Categorical comparison, optionally with error bars
Box / violin	Distribution across groups or timepoints
Scatter plot	Relationship between two continuous variables
Pie chart	Proportional composition of a whole
Spider web	Multivariate profile across radial axes
Flowchart	Patient disposition and enrollment flow
Multi-panel	Several endpoints shown as linked subplots

Rendering engines. Table 5 lists the chart types supported by each engine. Each figure is rendered using an engine that supports its type, with additional variation in visual style. Engine selection changes the appearance of the chart without changing the underlying data record.

Table 5: Chart types supported by each backend.

Backend	Supported chart types
Matplotlib	all 19
Seaborn	bar, box/violin, heatmap, scatter, line, volcano
Plotly	bar, scatter, line, waterfall, pie, box/violin, PK curves, oncology spider
plotnine	bar, scatter, line, box/violin, forest, volcano, waterfall, dose–response

Difficulty presets. Table 6 specifies the four difficulty presets. The presets jointly vary chart density, typography, annotations, and axis properties, subject to the characteristics of each chart type. As described in Section 3.2, these variations increase visual difficulty while keeping the figures within realistic clinical conventions.

Table 6: Difficulty presets. Each tier scales all properties together.

Property	Easy	Moderate	Challenging	Hard
Series / groups	2–3	3–4	4–6	5–8
Minimum font size	12	9	7	6
Legend columns	1	2	3	3
Log-scale axes	no	no	yes	yes
Annotation density	low	medium	high	very high
Unit complexity	standard	moderate	complex	ambiguous

B CAPE Implementation Details

B.1 Tool Outputs and Reliability Signals

CAPE's 13 vision tools operate on visual primitives shared across chart types. Table 7 lists the operation performed by each tool and the reliability signal it returns. These signals describe properties of the measurement, such as calibration residuals, curve coverage, and marker consistency. The agent uses them to identify readings that require further inspection or remeasurement, following the protocol in Section 4.2.

Calibration reports how tightly its ruler fits the anchors and whether a real tick was confirmed at each one. A curve trace reports how much of the line it followed and flags any abrupt jump that betrays a crossing onto a neighboring curve. A marker locator reports how many markers it found and whether the color groups separated cleanly. A heatmap reader reports its confidence in the detected grid. Such evidence lets the agent weigh a reading before committing the value. A flagged jump in a trace, for instance, prompts a re-seed on the intended line.

Table 7: The precision vision tools and the reliability signals they return to support verification.

Tool	Measures	Reliability signal
Calibrate	Pixel-to-value ruler for both axes	Fit residual; per-anchor confirmation
Trace curve	Points along a whole curve	Coverage; flagged jumps
Count lines	Number of distinct same-color lines	Spread of per-column counts
Refine polyline	Snaps a rough curve onto the ink	Points moved; mean shift
Read boxes	Quartiles and median of each box	Per-box coverage
Read error bars	Low and high ends of each whisker	Per-whisker coverage
Locate markers	Coordinates of every marker of a series	Count; size consistency
Locate groups	Per-group marker clouds in a scatter	Count; group separation
Read bars	Value of each bar from its baseline	Number of bars found
Read forest rows	Estimate and interval for each row	Number of rows found
Measure radial	Value at each spoke of a radar chart	Spoke coverage
Measure wedges	Angular fraction of each pie slice	Wedge count; fractions sum to 100%
Measure colormap	Heatmap cell values via the colorbar	Grid confidence

Anchor refinement. Algorithm 1 details the refinement of the value-location anchors supplied by the agent. Candidate tick locations are checked against both the supplied anchors and an inferred spacing model. When a candidate fails these checks, the original location is retained. The refined anchors determine the pixel-to-value mapping, while the fit residual and per-anchor confirmation provide evidence for the agent's subsequent verification.

B.2 Measurement Cases and Safeguards

The silent ruler. Every measured quantity rests on the ruler that maps pixels to data coordinates. Its correctness is therefore foundational. Earlier chart-extraction pipelines built this ruler by detecting tick pixels and matching them to axis values by their position along the axis (Luo et al., 2021; Rohatgi, 2024). A single miscounted tick can shift the entire scale without providing a signal that the calibration is incorrect. CAPE takes a different route and builds the ruler from anchors the agent has already matched. For each axis the agent supplies a handful of value-location pairs, from which

Algorithm 1 Anchor refinement for one axis.

Input: agent anchors $\{(v_k, p_k)\}$; search window w
Fit even-spacing model $\hat{p}(v)$ from $\{(v_k, p_k)\}$; mark trusted if the anchors lie on a line
for each anchor (v_k, p_k) **do**
    Detect candidate tick features within w of p_k
    $c \leftarrow$ nearest detected feature
    if c lies within w of p_k **and** within w of $\hat{p}(v_k)$ **then**
        accept c as the refined location
    else
        keep p_k ▷ never substitute the model estimate
    end if
end for
Fit $\text{value} = a\,\text{pixel} + b$ (in $\log$ space on a log axis) over the refined anchors
return ruler, per-anchor confirmation, fit residual

a least-squares fit recovers the mapping, cast in logarithmic form on a log axis. Because each value arrives already bound to its pixel, the fit needs no positional guesswork. A poor anchor then betrays itself by inflating the fit residual rather than corrupting the scale unseen. Each anchor is afterward sharpened locally, as summarized in Algorithm 1. Near the agent's rough mark the tool searches a small window for the true tick. It weighs each candidate against an even-spacing model inferred from the agent's own anchors. A candidate is adopted only when it agrees with both the agent's reading and this model. Absent such agreement, the tool holds to the agent's original mark rather than to a predicted position. Refinement can thereby tighten a sound anchor without unsettling one that is already correct. Two safeguards complete the ruler. A collision check flags any two distinct values that fall on a single pixel, an arrangement no genuine axis admits. When the axis occupies only a cramped crop, the region is enlarged before the agent aims, to keep closely spaced ticks from merging; the coordinates are mapped back to the original image once the reading is done.

Dense point clouds. Overlapping markers merge under a naive count. The marker locator separates a dense cloud into distinct points. The reported count then reflects the markers actually drawn.

Overlapping same-color lines. When several lines share a color and cross, color alone can no longer tell them apart. The tracer follows one line through its crossings from the point the agent indicates. Where no color boundary intervenes, a rough reading can be snapped onto the true ink.

Color bleed. Colors that look alike to a language model often diverge in the pixels. A value keyed to a named color can therefore bleed into a neighboring one. CAPE avoids naming color altogether. The agent marks a few points on the element; the tool then reads the true ink color at those pixels and adopts the shade on which they agree. A single stray point cannot mislead the reading, since the others outvote it.

B.3 Provenance and Self-checks

The verification process in Section 4.3 uses value provenance and checks that do not require ground truth. The following details describe how values are traced to their sources and checked against the chart.

Provenance. Every written value carries a record of its origin. This record marks the value as printed text, a tool measurement, or an inference. It further names the tool that produced the value and the region it came from. Such provenance serves two ends. It makes the extraction auditable, since any value can be traced to its source. It also informs the completeness check, which reads a field's provenance to separate a deliberate empty from one never measured. Provenance extends to geometry as well. Each reading region stores its offset in the original image together with its own ruler. A value measured within a cropped region can therefore be located on the full figure. This in turn lets the extraction be redrawn over the original chart for inspection.

Self-checks. Two further checks examine the finished record without any ground truth. The first is a landmark check. It compares the extracted values against anchors the figure fixes on its own, such

as a survival curve that begins at one, a proportion that stays within its bounds, or pie slices that sum to a hundred. These anchors owe nothing to the fitted ruler. A uniform shift that a ruler-based check would miss thus becomes visible. The second is a round-trip check. It sends a handful of extracted values back through the ruler that produced them and asks whether they land on the marks they came from. A ruler that has degenerated betrays itself here, when a value projects far from its ink or onto the wrong position. Such a fault can survive a small fit residual, yet the round trip brings it to the surface. Together these checks give the run a way to judge its own correctness with no reference values at hand, catching the kind of error that would otherwise pass unnoticed.

B.4 Future Work

A natural extension is to let CAPE learn from extraction failures and improve its measurement strategy across runs. Tool-derived reliability signals and record-level verification could provide feedback for learning when to invoke a tool, where to aim it, and when to repeat a measurement. Our synthetic generator could supply paired charts and ground truth for training and evaluation. A central challenge is to learn strategies that generalize across chart types rather than accumulate rules tailored to individual failures. Future work could investigate such policy learning while retaining deterministic verification, with evaluation on held-out chart types and rendering styles to assess generalization.

B.5 Worked Extraction Example

Figure 3 follows a Kaplan–Meier extraction through ten steps within the five stages of Section 4.1. Classification and the initial survey establish the schema and reading regions. Calibration maps pixel positions to axis values. After the initial trace reports abrupt jumps, the agent supplies rough polylines that the vision tool refines against the curve pixels. Printed annotations are read directly. The resulting values populate the schema before the run proceeds to verification.

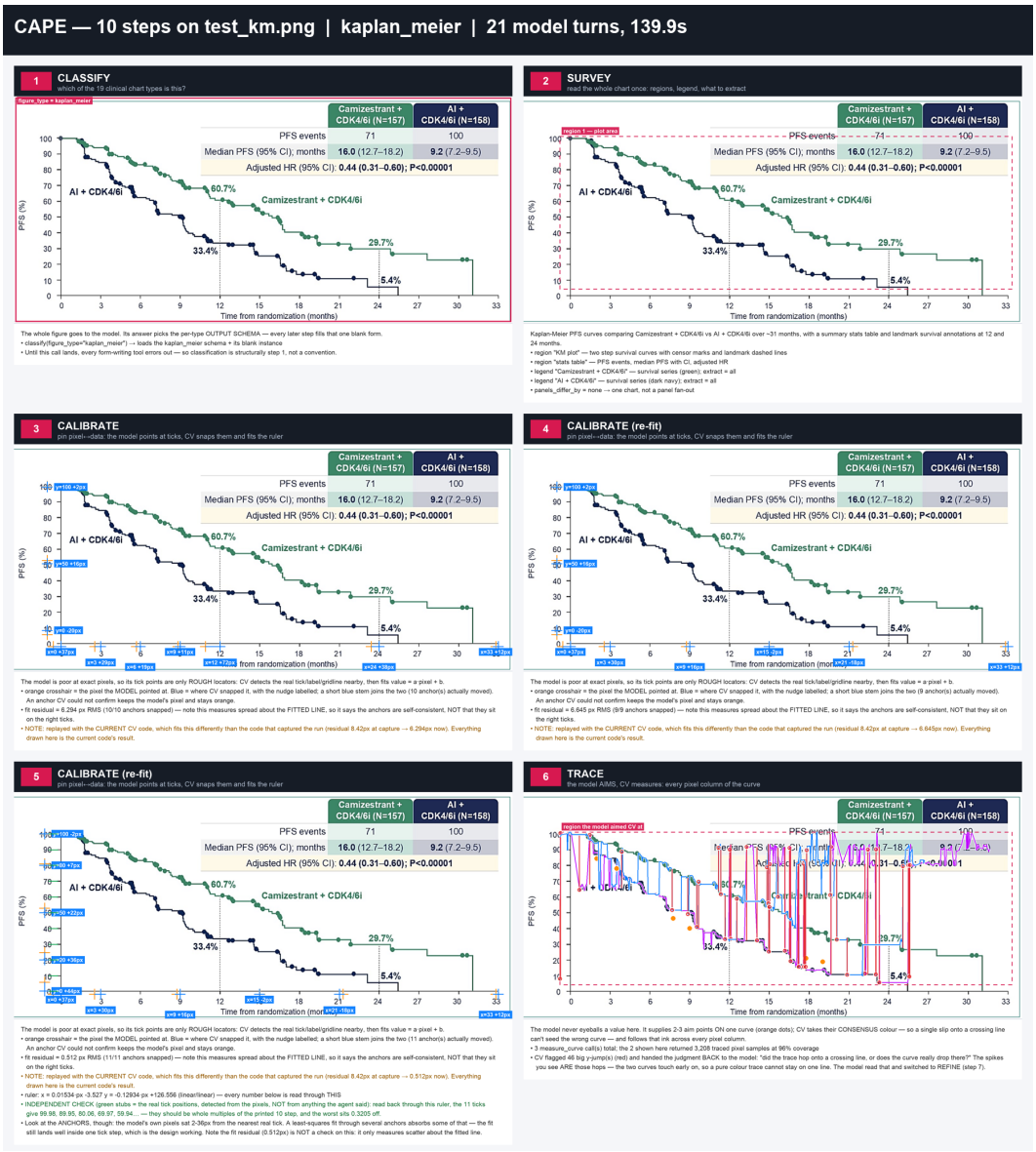


Figure 3: Worked Kaplan–Meier extraction, steps 1–6: classification, survey, repeated calibration, and initial tracing. Agent-supplied marks guide vision measurements; tracing feedback identifies abrupt jumps for further inspection. The calibration overlays are replayed with updated CV code, as noted in the panels, and differ from the originally recorded outputs.



Figure 3: Worked Kaplan–Meier extraction (continued), steps 7–10: polyline refinement, direct reading of printed values, schema filling, and verification.

C Evaluation Details

The evaluation in Section 5.1 scores field presence, textual labels, and numerical values separately. The definitions below specify the component scores and the treatment of arrays.

The first axis is structure. It asks whether the record is shaped correctly, that is, whether it holds the fields the ground truth defines. We measure it as the fraction of ground-truth top-level fields that also appear in the prediction,

$$\text{structure} = \frac{|K_{\text{gt}} \cap K_{\text{pred}}|}{|K_{\text{gt}}|}, \tag{3}$$

where K_{gt} and K_{pred} are the sets of top-level field names in the ground truth and the prediction. Structure checks only that a field is present, not that its contents are right. Those contents are the concern of the two remaining axes.

The second axis is labels. Every field whose value is text counts as a label, whether a title, an axis name, a category, or a legend entry. Each is scored by token-level F1 against the ground truth. A reading thus earns credit for the right words even when their order or casing differs. For a single string with predicted and true token sets P and T,

$$\text{F1} = \frac{2\,|P \cap T|}{|P| + |T|}. \tag{4}$$

The labels score is the mean of these per-field scores.

The third axis is numeric. It receives the largest weight to emphasize numerical accuracy. Following prior chart-extraction evaluation based on relative numerical error (Liu et al., 2023a), we score a predicted number against its ground-truth value as

$$s(\hat{v}, v) = \begin{cases} 1, & |v| < \epsilon \text{ and } |\hat{v}| < \epsilon, \\ 0, & |v| < \epsilon \text{ and } |\hat{v}| \geq \epsilon, \\ \max\left(0, 1 - \dfrac{|\hat{v} - v|}{|v|}\right), & |v| \geq \epsilon, \end{cases} \tag{5}$$

where $\hat{v}$ is the predicted value, v is the ground-truth value, and $\epsilon = 10^{-10}$. When $|v| < \epsilon$, the score is one if $|\hat{v}| < \epsilon$ and zero otherwise. For $|v| \geq \epsilon$, an exact prediction scores one, while an absolute error equal to or greater than $|v|$ scores zero. Most fields hold arrays rather than single numbers, for which order should not matter. Before scoring, predicted and ground-truth elements are matched by the Hungarian algorithm (Kuhn, 1955). A correct set of values is therefore credited regardless of its ordering. The matched scores are then divided by the length of the longer array,

$$\text{array} = \frac{\sum_{(i,j) \in M} s(\hat{v}_i, v_j)}{\max(n_{\text{pred}}, n_{\text{gt}})}, \tag{6}$$

where M is the set of matched index pairs and n_{pred}, n_{gt} are the two array lengths. Dividing by the longer length penalizes a record that omits points or invents them. An unmatched element adds nothing to the numerator yet still counts in the denominator.

The numeric axis weights each field by how much data it carries, so a dense array contributes more than a lone scalar. The component scores are combined using Equation 1 with the weights specified in Section 5.1.

Two safeguards keep the comparison fair. Field names are reconciled before scoring. A record that names the treatment arms differently from the ground truth is therefore not punished for the naming choice alone. Fields that belong only to the generator, which no reader could recover from the figure, are excluded.

D Detailed Results

Table 8 reports Overall scores for all 19 chart types on the synthetic benchmark. The comparison uses the Claude Agent SDK baseline with the same backbone as CAPE. The final column gives the difference between their scores, complementing the aggregate results in Section 5.3.

CAPE scores higher on 17 of the 19 chart types, with the largest gains on multi-panel, spider web, and PK figures. The per-type results also show variation that is not apparent from the aggregate score, including lower scores on dose–response and exposure–response charts.

Table 8: Per-type Overall score on the synthetic benchmark. Δ is CAPE minus Agent SDK. Higher score per row in bold.

Chart type	**Agent SDK**	**CAPE**	**Δ**
Longitudinal	0.876	**0.923**	$+0.047$
Spider web	0.678	**0.918**	$+0.240$
Multi-panel	0.531	**0.911**	$+0.380$
Swimmer	0.777	**0.898**	$+0.121$
Box / violin	0.805	**0.859**	$+0.054$
Pie chart	0.783	**0.849**	$+0.066$
Waterfall	0.829	**0.842**	$+0.013$
Kaplan–Meier	0.682	**0.822**	$+0.140$
Dose–response	**0.829**	0.814	-0.015
Bar plot	0.757	**0.796**	$+0.039$
Forest plot	0.749	**0.773**	$+0.024$
Scatter plot	0.634	**0.747**	$+0.113$
Heatmap	0.734	**0.735**	$+0.001$
Line plot	0.529	**0.728**	$+0.199$
PK curves	0.490	**0.695**	$+0.205$
Exposure–response	**0.709**	0.681	-0.028
Flowchart	0.510	**0.662**	$+0.152$
Volcano	0.531	**0.584**	$+0.053$
Oncology spider	0.488	**0.571**	$+0.083$
ALL	0.680	**0.780**	$+0.100$