Self-Ensembling Vision-Language Models for Chart Data Extraction

Anonymous ACL submission

Abstract

Charts effectively convey quantitative information, but the underlying data are often locked in image form, hindering reuse and analysis. Manually digitizing charts is time-consuming and error-prone, motivating automatic chart-to-table extraction. Recent approaches use specialized vision-language models (VLMs), yet performance still lags on charts with many datapoints or substantial stylistic variation. We propose a VLM self-ensembling method that repeatedly samples multiple tabular outputs from the same VLM for a fixed chart image and aggregates them at the level of individual table cells. We align candidate tables and take per-cell medians over numerical values to produce a more accurate consensus table. Our method also includes convergence detection to stop sampling once the aggregated table stabilizes, and uncertainty estimation based on dispersion across samples to help users assess extraction reliability. Because existing chart extraction benchmarks contain relatively simple plots with limited room for improvement, we introduce WB-ChartExtract, a new benchmark built from World Bank data with more complex and stylistically diverse charts; on average, its charts contain 8.4× more datapoints than those in the ChartQA benchmark. Across both ChartQA and WB-ChartExtract, our approach consistently improves extraction accuracy over single-pass VLM outputs, yielding up to 23% relative improvement on WB-ChartExtract after ensembling. More broadly, our method helps unlock tabular data previously siloed in chart images, enabling downstream analysis and reuse.

1 Introduction

Charts and plots are a primary medium for communicating quantitative information in scientific articles, policy reports, news stories, and online dashboards. Unlike tables, charts typically appear only as rasterized images in PDFs or on the web, making the underlying numbers difficult to access: they cannot be searched, joined with other sources, or reused for downstream analysis. Recovering these data at scale is increasingly important for meta-analysis, model validation, and real-time monitoring, where source data are often missing or inaccessible.

Manual digitization with tools such as WebPlotDigitizer (Rohatgi) is possible but slow and error-prone, especially for large corpora or visually complex figures. This has motivated automatic chart-to-table extraction, where a system reconstructs a structured table of datapoints from a chart image. Early systems (Leow et al., 2005; Savva et al., 2011) used engineered pipelines combining computer vision, optical character recognition (OCR), and heuristic reasoning about axes and legends. More recent work (Zhang et al., 2024; Chen et al., 2024; Xia et al., 2025) leverages vision-language models (VLMs) that can directly "read" charts and emit a textual table representation. While VLM-based methods simplify the stack and have improved performance and generalization on several benchmarks (Masry et al., 2022; Methani et al., 2020), they still struggle with dense datapoints, multiple overlaid series, or idiosyncratic labeling.

These failure modes reflect a broader pattern in generative models: a single forward pass can be brittle (Wang et al., 2023; Snell et al., 2024). When prompted multiple times on the same chart, a VLM can return noticeably different tables—dropping or adding points or entire series, or misreading values—yet these outputs are often partially correct and complementary. Existing chart extraction methods typically ignore this variability, relying on a single "best guess," and they provide little signal about extraction reliability, which is problematic for realistic, cluttered figures.

We propose a complementary approach that embraces VLM stochasticity by ensembling outputs at the level of individual table cells. Our method is

model-agnostic, can be layered on top of existing chart extraction systems, and applies to any chart type the underlying model can handle. Concretely, we repeatedly prompt a VLM on the same chart image, align candidate tables at the cell level, and compute per-cell medians over numerical values to form a consensus table. We also introduce a convergence detection mechanism to stop sampling once additional predictions are unlikely to change the ensemble, and an uncertainty estimate based on variability across samples to quantify extraction reliability.

Progress on chart extraction also depends on benchmarks with challenging, stylistically varied charts. However, widely used datasets such as ChartQA offer increasingly limited room for improvement: state-of-the-art VLMs already achieve high performance, in part because many plots are relatively simple, with few datapoints, limited stylistic variation, and numbers printed directly on the chart. To provide a more demanding testbed, we introduce WB-ChartExtract, a benchmark of synthetic charts constructed from real World Bank data. WB-ChartExtract spans many countries, diverse economic and social indicators, and a variety of plotting styles. On average, its charts contain 8.4× more datapoints than ChartQA charts and feature multiple overlaid series and long time horizons, making them substantially more challenging.

We evaluate our self-ensembling approach on both ChartQA and WB-ChartExtract. Across datasets and underlying models, ensembling consistently improves accuracy over single-pass VLM extraction, with particularly strong gains on WB-ChartExtract (up to 23% relative improvement after ensembling). We further analyze which error types our approach corrects and which remain challenging. Importantly, we show that convergence detection is well calibrated and supports a controllable trade-off between computational cost and marginal accuracy gains via practical early stopping. Finally, we demonstrate that our uncertainty estimate is empirically inversely correlated with extraction accuracy.

Our contributions are fourfold:

1. A model-agnostic self-ensembling method for chart data extraction.

2. A convergence detection mechanism that identifies when additional samples are unlikely to change the aggregated result.

3. Ensemble uncertainty estimates to help users gauge extraction reliability.

4. WB-ChartExtract, a new benchmark constructed from real-world data that is, to our knowledge, the most challenging to date.

Together, these contributions move chart-to-table extraction closer to real-world deployment, helping unlock structured data currently trapped in chart images and enabling richer downstream quantitative analysis.

2 Background and Related Work

Chart data extraction (also called *chart-to-table* or *derendering*) aims to recover the numerical values in a chart image along with their semantic structure (e.g., axis values and series names). Early systems decomposed the task into multi-stage pipelines combining computer vision, OCR, and heuristics to map pixels to values (Leow et al., 2005; Savva et al., 2011; Mishchenko and Vassilieva, 2011; Al-Zaidy and Giles, 2017). While effective when figures match anticipated templates, these pipelines are often brittle under modest stylistic variation (e.g., layout, font, or color).

Recent work instead uses VLMs to generate a textual table representation directly from the chart image (Liu et al., 2023b,a; Han et al., 2023; Meng et al., 2024; Chen et al., 2024; Zhang et al., 2024; Xia et al., 2025). However, VLMs often struggle with fine-grained visual distinctions (Liu et al., 2025; Chen et al., 2025), including in chart settings (Razeghi et al., 2024). Consequently, one-pass generation remains error-prone on complex charts: models may omit datapoints, hallucinate values, misread axes, or inconsistently name series across runs. Our approach is designed to sit atop either specialized chart extraction models or generalist VLMs, repeatedly querying the underlying model and aggregating outputs to improve robustness.

Benchmarks. Common chart-extraction datasets include ChartQA (Masry et al., 2022), PlotQA (Methani et al., 2020), and ChartX (Xia et al., 2025). ChartQA contains real-world charts, but many instances are visually simple (e.g., values printed directly on bars). PlotQA and ChartX provide large-scale synthetic charts with broader chart-type coverage, but limited stylistic diversity and complexity. In contrast, WB-ChartExtract does not print values directly on marks, includes longer

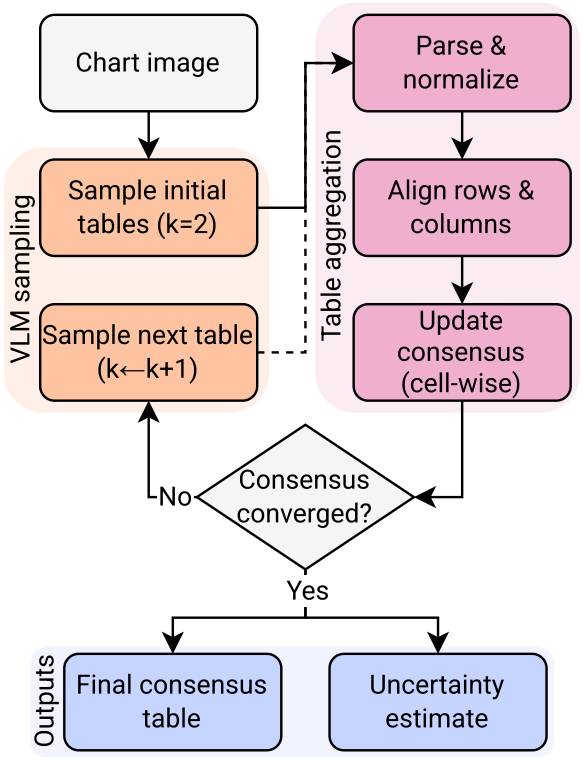

Figure 1: Iterative self-ensembling for chart-to-table extraction. We sample tables from a base VLM, parse/normalize and align rows/columns, and update a cell-wise consensus until convergence, producing a final table and uncertainty estimate based on MAD.

series and multiple series per chart, and exhibits substantial stylistic variance.

3 Methodology

Given a chart image, our goal is to recover the underlying table of numeric datapoints. Our approach is *model-agnostic*: we repeatedly prompt a base VLM to produce tabular outputs, then align and ensemble these outputs at the level of individual table cells to obtain a consensus table. Compared to single-pass extraction, self-ensembling suppresses outlier predictions (e.g., hallucinated or missing datapoints) and yields a natural uncertainty signal. Figure 1 provides an end-to-end overview.

3.1 Base Model Sampling

For each chart image, we query a base VLM (by default, Llama 4 Maverick(Meta AI, 2025)[1]) with the following prompt:

> Here is an image of a chart. Please extract the numerical data it represents and return it in TSV (tab-separated values) format with appropriate headers. Copy the headers exactly as they are in the image. IMPORTANT: For the TSV, use tab (`\t`) as the separator. Remember: The sole output should be the TSV table surrounded by "'tsv "'. Nothing else.

We sample at temperature T (a hyperparameter; Section 4.7). For each image, we generate two initial tables, parse, align, and aggregate them (as described below). We continue sampling and updating the ensemble until either: (i) for *patience*$= 2$ consecutive updates, at least *coverage*$= 95\%$ of aggregated cell values change by no more than a relative *tolerance*$= 10\%$ between successive ensemble updates (sensitivity analysis in Section 4.6.3); or (ii) a maximum number of samples $K_{\max}$ is reached. We use $K_{\max} = 20$ by default based on our computational budget, but this can be adjusted (Section 4.6.2).

3.2 Parsing and Normalization

Each sample is a TSV-like text block and may contain formatting errors (e.g., inconsistent column counts). We first parse as TSV; if parsing fails due to ragged rows, we repair by padding or truncating rows to the modal column count.

After parsing, we treat (i) the first row as column headers, (ii) the first column as row headers (index), and (iii) remaining cells as numeric values. We convert value cells to numeric, stripping common artifacts (commas, currency symbols, percent signs); non-numeric strings (including empty cells) are treated as missing and set to `nan`. If needed, we transpose the table to enforce more rows than columns, which simplifies alignment for time-on-x charts with multiple series.

3.3 Row and Column Alignment via Clustering

Across samples, tables may differ in row/column order and labels may be noisy (e.g., spelling, abbreviations). We therefore align tables by clustering row and column labels (independently) across samples to form canonical row and column groups.

We measure label similarity with ANLS (Biten et al., 2019, average normalized Levenshtein similarity), yielding $\mathrm{sim}(\ell_1, \ell_2) \in [0, 1]$ (Liu et al., 2023a; Furkan Biten et al., 2019). Clusters must satisfy: (i) grouped labels are similar under sim, and (ii) no cluster contains two labels from the same sampled table (to avoid merging within-table duplicates).

[1]We developed primarily with Llama 4 Maverick due to its efficiency and low inference cost on Groq (Groq, Inc., 2025), but the method is model-agnostic.

We use greedy clustering with threshold $\tau = 0.5$ (following Liu et al. (2023a); Furkan Biten et al. (2019)). Each cluster c is represented by $\mathrm{rep}(c)$, the first label assigned to c:

1. Initialize an empty set of clusters.
2. For each label ℓ, assign it to an existing cluster c if (i) $\mathrm{sim}(\ell, \mathrm{rep}(c)) \geq \tau$ and (ii) c contains no label from the same sampled table as ℓ. If multiple clusters qualify, choose the one maximizing $\mathrm{sim}(\ell, \mathrm{rep}(c))$.
3. If no cluster qualifies, create a new cluster containing ℓ.

We then prune clusters that appear in fewer than a specified proportion of tables (a hyperparameter; Section 4.8) to remove spurious labels that would otherwise introduce extra rows/columns. For each remaining cluster, we define its canonical label as the most frequent label string in the cluster (ties broken at random) and use these canonical labels as the aggregated table's row/column names.

3.4 Cell-Wise Aggregation

Let $\mathcal{R} = \{r_1, \ldots, r_{|\mathcal{R}|}\}$ and $\mathcal{C} = \{c_1, \ldots, c_{|\mathcal{C}|}\}$ be the retained row and column clusters. For each aligned cell $(r, c) \in \mathcal{R} \times \mathcal{C}$, we collect all numeric predictions from sampled tables whose row label falls in r and column label falls in c. Let $V_{r,c}$ denote this set (ignoring `nan`). We aggregate as $\hat{y}_{r,c} = \mathrm{median}(V_{r,c})$ (ablation in Appendix A); if $V_{r,c}$ is empty, we output `nan`. The final table orders row and column clusters by lexicographic sort of their canonical labels.

3.5 Uncertainty Estimation

Self-ensembling yields a natural uncertainty signal from disagreement across samples: consistent values imply reliable medians, while high variability indicates brittle extraction and additional verification (or more samples) may be warranted.

For each aligned cell (r, c), we compute uncertainty as the median absolute deviation (MAD) over $V_{r,c}$, normalized by the magnitude of the ensembled value:

$$u_{r,c} = \frac{\mathrm{MAD}_{r,c}}{|\hat{y}_{r,c}|},$$

$$\text{where} \quad \mathrm{MAD}_{r,c} = \mathrm{median}_{v \in V_{r,c}} |v - \hat{y}_{r,c}| .$$

To avoid division by zero, we exclude cells with $\hat{y}_{r,c} = 0$ when computing table-level uncertainty summaries. Relative MAD is scale-invariant, making uncertainty comparable across charts and across cells with different magnitudes.

To obtain a single uncertainty score per chart, we summarize $\{u_{r,c}\}$ over cells with non-missing predictions and $\hat{y}_{r,c} \neq 0$ using: U_{med} (median), U_{mean} (mean), and $U_{\max}$ (maximum). These capture complementary failure modes: U_{med} reflects typical uncertainty, U_{mean} reflects overall dispersion, and $U_{\max}$ flags catastrophic disagreement in any cell.

4 Experiments

4.1 Datasets

ChartQA. We evaluate on ChartQA, the most widely used benchmark for chart data extraction. Its test set contains 1,509 real-world chart images, but has notable limitations: charts are often visually simple, with relatively few datapoints, limited stylistic diversity, and values frequently printed directly on the chart, allowing models to rely on text rather than chart geometry and axes. As a result, the benchmark is somewhat saturated. ChartQA's ground-truth tables are also often noisy, reducing evaluation reliability.

WB-ChartExtract. To address these limitations, we introduce WB-ChartExtract, a new benchmark constructed from World Bank Data Bank time series (The World Bank) spanning 52 indicators, 218 countries, and 65 years. We prune series with more than half missing values or where there are missing values in the interior (i.e., not just at the ends) of the series. We then create 1,000 datasets by repeatedly sampling one indicator and 2–3 countries at random and collecting the corresponding series, using each underlying series at most once. Each dataset is rendered as a Matplotlib line chart (Hunter, 2007), randomizing font family and size, grid presence and style, plotting style, markers, line styles, colors, line widths, transparency, and figure size. This yields 1,000 charts with diverse visuals and clean ground-truth tables. On average, WB-ChartExtract charts contain 8.4× more datapoints than ChartQA charts, and none include printed numeric value labels. Figure 2 shows example charts.

4.2 Evaluation Metric

We evaluate chart data extraction using *Relative Mapping Similarity* (Liu et al., 2023a, RMS), a table-matching metric that treats a table as an unordered set of mappings from a (row header, col-

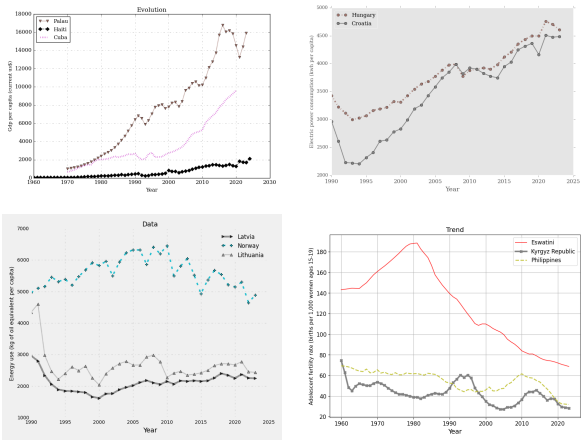

Figure 2: Examples of charts in WB-ChartExtract, our benchmark of more complex and stylistically diverse charts.

umn header) pair to a cell value. We represent the predicted table as $P = \{p_i\}_{i=1}^{N}$ and the ground-truth table as $T = \{t_j\}_{j=1}^{M}$, where each entry is a triple $p_i = (p_i^r, p_i^c, p_i^v)$ and $t_j = (t_j^r, t_j^c, t_j^v)$. This set representation is invariant to row/column permutations.

RMS compares *keys* (row/column headers) using a thresholded normalized Levenshtein distance $\mathrm{NL}_\tau(\cdot,\cdot)$, clipping distances above τ to 1. Keys are formed by concatenating headers: $k(p_i) = p_i^r \| p_i^c$ and $k(t_j) = t_j^r \| t_j^c$. Numeric values are compared by a clipped relative error

$$D_\theta(p,t) = \min\Big(1, \frac{\|p-t\|}{\|t\|}\Big),$$

and combined into an entry-level similarity

$$s(p_i,t_j) = \Big(1-\mathrm{NL}_\tau(k(p_i),k(t_j))\Big)\Big(1-D_\theta(p_i^v,t_j^v)\Big),$$

which is high only when both headers and values agree.

RMS then finds a minimum-cost bipartite matching between predicted and target entries (key-based), yielding an assignment matrix $X \in \mathbb{R}^{N\times M}$, and computes mapping-level precision and recall:

$$\mathrm{RMS}_{\mathrm{prec}} = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} X_{ij}\, s(p_i,t_j),$$

$$\mathrm{RMS}_{\mathrm{rec}} = \frac{1}{M}\sum_{i=1}^{N}\sum_{j=1}^{M} X_{ij}\, s(p_i,t_j).$$

We report RMS_{F1}, the harmonic mean of $\mathrm{RMS}_{\mathrm{prec}}$ and $\mathrm{RMS}_{\mathrm{rec}}$. Following Liu et al. (2023a), we set $\tau = 0.5$ and $\theta = 0.1$.

Method	ChartQA	WB-ChartExtract
OneChart	35.93	16.24
TinyChart	95.20	16.80
+ Self-ens. (ours)	95.28	18.44
Qwen3-VL 235B	91.43	35.07
+ Self-ens. (ours)	93.21	43.19
Llama 4 Maverick	83.35	47.12
+ Self-ens. (ours)	86.47	51.72
GPT-5.1	84.80	59.07

Table 1: Main results ($\mathrm{RMS}_{\mathrm{F1}}$; higher is better). "Self-ens." denotes our self-ensembling procedure applied on top of a base model's direct predictions.

Because predictions may be transposed, RMS also scores both the predicted table and its transpose and takes the higher RMS_{F1}, yielding transposition invariance.

4.3 Main Results

We compare against single-pass predictions from two chart-specialized models (OneChart (Chen et al., 2024), TinyChart (Zhang et al., 2024)), two open-source VLMs (Qwen3-VL 235B A22B Instruct (Bai et al., 2025), Llama 4 Maverick (Meta AI, 2025)), and one closed-source VLM (GPT-5.1 (OpenAI, 2025)). To demonstrate model-agnosticism, we apply self-ensembling on top of TinyChart, Qwen3-VL 235B A22B Instruct, and Llama 4 Maverick. For convergence detection, we use default parameters with a tolerance of 1%.

Table 1 reports $\mathrm{RMS}_{\mathrm{F1}}$ on ChartQA and WB-ChartExtract. Self-ensembling consistently improves over the corresponding single-pass outputs across all three base models and both datasets. Gains are larger on WB-ChartExtract (+1.64 to +8.12 $\mathrm{RMS}_{\mathrm{F1}}$), suggesting aggregation is especially beneficial on more complex charts. Improvements on ChartQA are smaller but still consistent (+0.08 to +3.12), consistent with ChartQA offering less headroom.

Our best self-ensembled configurations reach 95.28 $\mathrm{RMS}_{\mathrm{F1}}$ on ChartQA (TinyChart + self-ensembling) and 51.72 on WB-ChartExtract (Llama 4 Maverick + self-ensembling). The latter improves over its direct-prediction counterpart by +4.60 points, though it remains below the best overall direct-prediction baseline, GPT-5.1 (59.07).

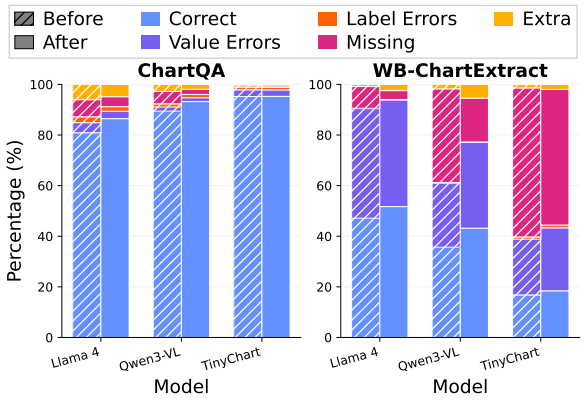

Figure 3: Error-type breakdown for each base model before vs. after self-ensembling, shown separately for ChartQA and WB-ChartExtract.

4.4 Error Analysis

To understand how self-ensembling improves RMS_{F1}, we decompose each model's error into five components that sum to 100%: **Correct** (example-level F1), **Value Errors** (numeric mismatch under the relative error distance), **Label Errors** (imperfect label matching under ANLS), **Missing** (ground-truth datapoints with no matched prediction), and **Extra** (predicted datapoints with no matched ground-truth). Figure 3 shows the average breakdown before and after ensembling.

On ChartQA, **Correct** is already high, leaving limited headroom. Ensembling primarily reduces **Missing** and, to a lesser extent, **Extra** for Llama 4 Maverick and Qwen3-VL, with modest decreases in **Value Errors**. **Label Errors** remain small throughout (all $\leq$2.31%), suggesting that omitted or spurious datapoints dominate over header mismatches.

WB-ChartExtract shows a different profile: **Label Errors** are negligible (0.13–1.10%), while errors are driven by **Missing** and especially **Value Errors**. Ensembling substantially reduces **Missing** across models, but can increase **Extra** (notably for Llama and Qwen), consistent with recovering additional candidates that do not correspond to ground truth on dense, multi-series charts.

On WB-ChartExtract, **Value Errors** remain the largest component and do not consistently decrease. This is expected: when **Missing** shrinks, more ground-truth datapoints become matched and therefore contribute numeric penalties, which can increase the **Value Errors** mass even as **Correct** improves. Overall, self-ensembling is most effective at recovering omitted datapoints, while accurately estimating their values remains the main challenge on WB-ChartExtract.

4.5 Empirical Validation of Uncertainty Estimation

We assess whether ensemble disagreement predicts extraction quality by correlating each table-level uncertainty summary with final RMS_{F1}. On ChartQA, relative MAD is strongly anti-correlated with RMS_{F1} (Spearman $\rho = -0.40$ for U_{med} and $\rho = -0.55$ for both U_{mean} and U_{max}; all $p < 0.001$). On WB-ChartExtract, correlations are stronger, with ρ ranging from -0.68 to -0.75 across summaries (all $p < 0.001$). Thus, higher ensemble disagreement reliably indicates lower extraction accuracy, making relative MAD a useful uncertainty estimate for downstream decisions.

The tighter relationship on WB-ChartExtract likely reflects higher chart complexity: errors more often stem from ambiguous visual inference (e.g., reading values from axes and marks), producing divergent samples and elevated dispersion. On ChartQA, where many values are printed as text, samples may agree even when they share a systematic mistake (e.g., consistent header mismatch), weakening the correlation.

4.6 Convergence Detection

4.6.1 Convergence Detection Ablation

We ablate convergence detection using Llama 4 Maverick. With convergence detection *off*, we always sample a fixed budget of $K_{\text{max}} = 20$ tables per image. With early stopping *on*, we stop once the ensemble meets our convergence criterion, up to the same K_{max}.

Table 2 shows that early stopping yields large compute savings while preserving most of the ensemble gain over the single-pass baseline. On ChartQA, fixed-budget ensembling improves RMS_{F1} from 83.35 to 87.01 (+3.66), while early stopping reaches 86.39 (+3.04), retaining **83%** of the ensembling boost with **4.47**$\times$ **fewer** samples on average (20.0$\rightarrow$4.47). On WB-ChartExtract, fixed-budget ensembling improves from 47.12 to 51.97 (+4.85), while early stopping reaches 50.85 (+3.73), retaining **77%** of the boost with **2.19**$\times$ **fewer** samples (20.0$\rightarrow$9.14). The higher $\bar{S}$ on WB-ChartExtract is consistent with its greater chart complexity, which requires more iterations for the ensemble to stabilize.

These results suggest convergence detection is practical for deployment, substantially reducing

Early stopping	ChartQA		WB-ChartExtract	
	RMS_{F1}	$\bar{S}$	RMS_{F1}	$\bar{S}$
Off (fixed budget)	87.01	20.0	51.97	20.0
On (early stopping)	86.39	4.47	50.85	9.14

Table 2: Effect of early stopping. $\bar{S}$ is the average number of samples to convergence per image (lower is better).

sampling cost while preserving most ensemble gains. Unless otherwise noted, we report results with early stopping on.

4.6.2 Ensemble Size

Figure 4 shows how RMS_{F1} and convergence evolve with ensemble size K (Llama 4 Maverick). RMS_{F1} improves quickly for small K and then shows diminishing returns: on ChartQA, RMS_{F1} rises sharply from $K{=}1$ to $K{=}5$ and largely saturates by $K{\approx}7$. WB-ChartExtract benefits longer, increasing gradually beyond $K{\approx}5$ before plateauing around $K{\approx}15$, consistent with its higher chart complexity.

Most examples converge quickly, but convergence is slower on WB-ChartExtract. On ChartQA, the converged fraction rises steeply and is near-saturated by $K{\approx}9$, with a median early-stopping point at $K{=}4$ (gray dashed line). On WB-ChartExtract, the converged fraction grows more slowly, remains below 90% at $K{=}20$, and has a later median stopping point at $K{=}7$. Thus, early stopping triggers quickly for most ChartQA examples and later for WB-ChartExtract, reflecting higher chart complexity.

4.6.3 Sensitivity to Convergence Hyperparameters

Our early-stopping rule declares convergence when, between two consecutive ensemble updates, at least a fraction *coverage* of aggregated cells change by no more than a relative *tolerance*, and this holds for *patience* consecutive updates. Increasing patience or coverage, or decreasing tolerance, makes the criterion stricter, typically requiring more samples but yielding marginal accuracy gains.

Tables 3–5 confirm this trade-off on ChartQA and WB-ChartExtract. Increasing **patience** (1→3) monotonically raises $\bar{S}$ while slightly improving RMS_{F1} (Table 3). Increasing **coverage** (90%→97.5%) similarly increases $\bar{S}$ with modest RMS_{F1} changes (Table 4). **Tolerance** has the largest compute impact: tighter tolerance (1%) substantially increases $\bar{S}$, while looser tolerance (50%) stops earlier but reduces RMS_{F1} (Table 5).

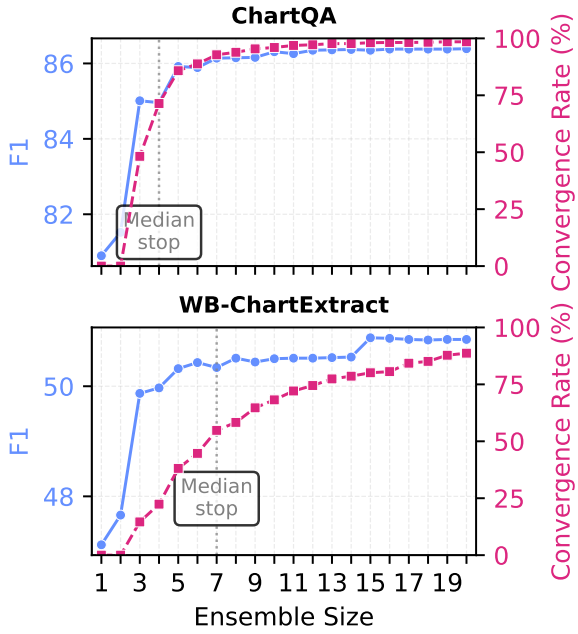

Figure 4: Effect of ensemble size K on RMS_{F1} and the percentage of examples that have met the convergence criterion by size K. Gray dashed lines mark the median stopping iteration under early stopping.

Patience	ChartQA		WB-ChartExtract	
	RMS_{F1}	$\bar{S}$	RMS_{F1}	$\bar{S}$
1	85.63	3.12	49.87	6.58
2 (default)	86.39	4.47	50.85	9.14
3	86.55	5.80	51.04	10.85

Table 3: Sensitivity to *patience*, fixing coverage=95% and tolerance=10%. $\bar{S}$ is the average samples per image.

These hyperparameters therefore tune the cost–accuracy frontier. Unless otherwise noted, we use patience= 2, coverage= 95%, and tolerance= 10%.

4.7 Sampling Temperature

Sampling temperature controls candidate-table stochasticity: lower T is more deterministic, while higher T increases diversity but can reduce per-sample reliability. For self-ensembling, this trades off individual accuracy against ensemble diversity.

We study this trade-off on ChartQA and WB-ChartExtract using Llama 4 Maverick, varying T and ensemble size. Figure 5 plots RMS_{F1} versus ensemble size for $T \in \{0.0, 1.0, 2.0\}$.

On ChartQA, lower temperatures perform best for small ensembles ($K \leq 2$): $T{=}0.0$ outperforms $T{=}1.0$ and $T{=}2.0$, so we use $T{=}0.0$ for

Coverage	ChartQA		WB-ChartExtract	
	RMS_{F1}	$\bar{S}$	RMS_{F1}	$\bar{S}$
90%	86.35	4.43	50.45	7.27
95% (default)	86.39	4.47	50.85	9.14
97.5%	86.41	4.61	50.94	10.89

Table 4: Sensitivity to *coverage*, fixing patience=2 and tolerance=10%. $\bar{S}$ is the average samples per image.

Tolerance	ChartQA		WB-ChartExtract	
	RMS_{F1}	$\bar{S}$	RMS_{F1}	$\bar{S}$
1%	86.47	5.14	51.72	16.43
10% (default)	86.39	4.47	50.85	9.14
50%	86.35	4.30	50.22	5.62

Table 5: Sensitivity to *tolerance*, fixing patience=2 and coverage=95%. $\bar{S}$ is the average samples per image.

direct-prediction baselines. For larger ensembles ($K \geq 5$), T=2.0 yields the best validation RMS_{F1}, indicating that added diversity outweighs reduced per-sample determinism; we therefore use T=2.0 for self-ensembling on ChartQA.

On WB-ChartExtract, T=1.0 is marginally best across ensemble sizes, with T=0.0 and T=2.0 nearly matching it for $K \geq 6$. We therefore use T=1.0 for both self-ensembling and direct-prediction baselines on WB-ChartExtract.

4.8 Cluster Pruning Threshold

We ablate the cluster pruning threshold, which sets the minimum fraction of sampled tables in which a row/column label must appear to be retained after clustering. Higher thresholds remove low-support (often spurious) labels, while lower thresholds retain more labels but risk noise. Using Llama 4 Maverick, we test thresholds in $\{0.0, 0.1, 0.2, 0.3, 0.4, 0.5\}$, where 0.0 denotes no pruning.

Table 6 reports RMS_{F1}. On ChartQA, performance increases monotonically with the threshold, suggesting aggressive pruning improves aggregation by removing noisy labels. On WB-ChartExtract, moderate pruning is best (0.1–0.2); more aggressive pruning slightly hurts by removing valid but less frequent labels. We therefore use a pruning threshold of 0.5 for ChartQA and 0.1 for WB-ChartExtract in all experiments.

5 Conclusion

We introduced a model-agnostic self-ensembling method for chart data extraction that exploits VLM stochasticity across repeated runs to produce more accurate consensus tables than a single pass. The method also provides ensemble uncertainty estimates and a practical convergence-based early-stopping strategy. To better stress-test chart extraction, we presented WB-ChartExtract, a new benchmark derived from World Bank data with more challenging charts, including multiple overlaid series, long time horizons, and greater stylistic variation than widely used datasets. Across ChartQA and WB-ChartExtract, self-ensembling consistently improves over single-pass VLM baselines.

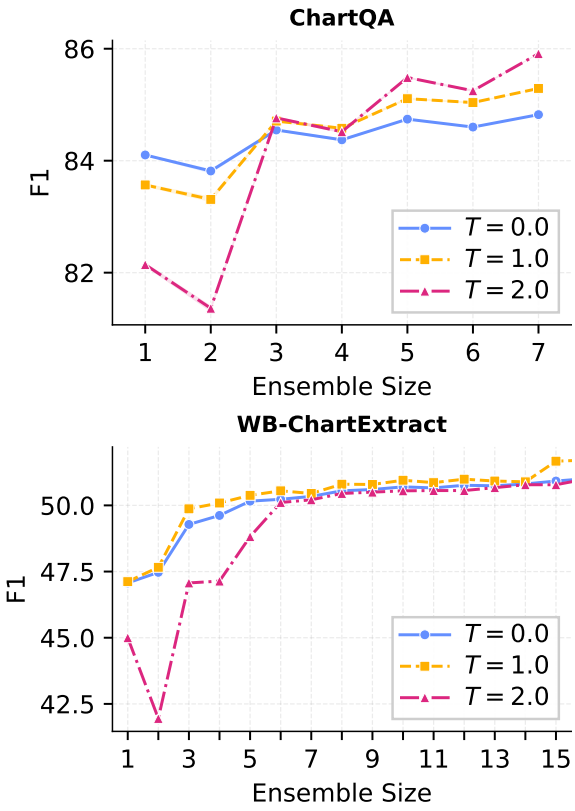

Figure 5: Effect of sampling temperature on ensemble performance using Llama 4 Maverick on ChartQA (top) and WB-ChartExtract (bottom).

Pruning Threshold	ChartQA	WB-ChartExtract
0.0 (no pruning)	79.14	50.62
0.1	81.37	**51.72**
0.2	82.91	51.69
0.3	84.63	51.22
0.4	86.01	50.90
0.5	**86.53**	50.19

Table 6: Effect of the cluster pruning threshold on RMS_{F1} using Llama 4 Maverick.

6 Limitations

Our approach inherits several limitations from both the underlying VLM and our ensembling procedure. First, self-ensembling is bounded by base-model failure modes: it can only aggregate among the candidate tables the VLM produces. When the model makes systematic errors, aggregation may not correct them and can even reinforce consistent mistakes.

Second, self-ensembling increases compute cost and latency. The method requires multiple VLM calls per chart, along with parsing, clustering, alignment, and aggregation. Although early stopping reduces the average number of samples, the overall pipeline remains slower and more expensive than single-pass extraction, which can limit throughput in large-scale deployments.

Third, our early-stopping criterion focuses on numeric stability and may miss "semantic" non-convergence. Because convergence is defined by relative changes in aggregated numeric cells, the ensemble may stabilize while the table structure remains incorrect, particularly when such structural mistakes are consistent across samples.

Fourth, our uncertainty estimate is heuristic and incomplete. Relative MAD provides a useful disagreement signal, but it is not a calibrated probability of correctness. In particular, uncertainty can be low even when the model is consistently wrong due to systematic errors, so it should be interpreted as a diagnostic rather than a formal confidence measure.

Fifth, WB-ChartExtract is synthetically rendered. While based on real World Bank time series, the charts are generated with Matplotlib, and may not capture real-world artifacts.

Finally, our empirical study does not fully cover the strongest available base models. We apply self-ensembling to TinyChart, Qwen3-VL, and Llama 4 Maverick, but not to GPT-5.1 (the best direct-prediction baseline on WB-ChartExtract) due to computational cost, leaving open how much self-ensembling can benefit the most capable systems.

References

Rabah A. Al-Zaidy and C. Lee Giles. 2017. A machine learning approach for semantic structuring of scientific charts in scholarly documents. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, page 4644–4649. AAAI Press.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang 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, Lluis Gomez, Marçal Rusiñol, Ernest Valveny, C. V. Jawahar, and Dimosthenis Karatzas. 2019. Scene text visual question answering. *Preprint*, arXiv:1905.13648.

Jinyue Chen, Lingyu Kong, Haoran Wei, Chenglong Liu, Zheng Ge, Liang Zhao, Jianjian Sun, Chunrui Han, and Xiangyu Zhang. 2024. Onechart: Purify the chart structural extraction via one auxiliary token. *Preprint*, arXiv:2404.09987.

Shiqi Chen, Tongyao Zhu, Ruochen Zhou, Jinghan Zhang, Siyang Gao, Juan Carlos Niebles, Mor Geva, Junxian He, Jiajun Wu, and Manling Li. 2025. Why is spatial reasoning hard for vlms? an attention mechanism perspective on focus areas. *Preprint*, arXiv:2503.01773.

Ali Furkan Biten, Rubèn Tito, Andrés Mafla, Lluis Gomez, Marçal Rusiñol, Minesh Mathew, C.V. Jawahar, Ernest Valveny, and Dimosthenis Karatzas. 2019. Icdar 2019 competition on scene text visual question answering. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1563–1570.

Groq, Inc. 2025. Groq is fast, low cost inference.

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

Peter J. Huber. 1964. Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, 35(1):73–101.

J. D. Hunter. 2007. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95.

Wee Kheng Leow, Weihua Huang, and Chew Lim Tan. 2005. Associating Text and Graphics for Scientific Chart Understanding . In *Proceedings. Eighth International Conference on Document Analysis and Recognition*, pages 580–584, Los Alamitos, CA, USA. IEEE Computer Society.

Chengzhi Liu, Zhongxing Xu, Qingyue Wei, Juncheng Wu, James Zou, Xin Eric Wang, Yuyin Zhou, and Sheng Liu. 2025. More thinking, less seeing? assessing amplified hallucination in multimodal reasoning models. *Preprint*, arXiv:2505.21523.

Fangyu Liu, Julian Eisenschlos, Francesco Piccinno, Syrine Krichene, Chenxi Pang, Kenton Lee, Mandar Joshi, Wenhu Chen, Nigel Collier, and Yasemin

Altun. 2023a. DePlot: One-shot visual language reasoning by plot-to-table translation. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 10381–10399, Toronto, Canada. Association for Computational Linguistics.

Fangyu Liu, Francesco Piccinno, Syrine Krichene, Chenxi Pang, Kenton Lee, Mandar Joshi, Yasemin Altun, Nigel Collier, and Julian Martin Eisenschlos. 2023b. Matcha: Enhancing visual language pretraining with math reasoning and chart derendering. *Preprint*, arXiv:2212.09662.

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. *Preprint*, arXiv:2203.10244.

Fanqing Meng, Wenqi Shao, Quanfeng Lu, Peng Gao, Kaipeng Zhang, Yu Qiao, and Ping Luo. 2024. Chartassisstant: A universal chart multimodal language model via chart-to-table pre-training and multitask instruction tuning. *Preprint*, arXiv:2401.02384.

Meta AI. 2025. The llama 4 herd: The beginning of a new era of natively multimodal intelligence. https://ai.meta.com/blog/llama-4-multimodal-intelligence/.

Nitesh Methani, Pritha Ganguly, Mitesh M. Khapra, and Pratyush Kumar. 2020. Plotqa: Reasoning over scientific plots. *Preprint*, arXiv:1909.00997.

Ales Mishchenko and Natalia Vassilieva. 2011. Chart image understanding and numerical data extraction. pages 115–120.

OpenAI. 2025. Gpt-5.1: A smarter, more conversational chatgpt. https://openai.com/index/gpt-5-1/.

Yasaman Razeghi, Ishita Dasgupta, Fangyu Liu, Vinay Venkatesh Ramasesh, and Sameer Singh. 2024. Plot twist: Multimodal models don't comprehend simple chart details. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 5922–5937, Miami, Florida, USA. Association for Computational Linguistics.

Ankit Rohatgi. Webplotdigitizer.

Manolis Savva, Nicholas Kong, Arti Chhajta, Li Fei-Fei, Maneesh Agrawala, and Jeffrey Heer. 2011. Revision: automated classification, analysis and redesign of chart images. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, page 393–402, New York, NY, USA. Association for Computing Machinery.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *Preprint*, arXiv:2408.03314.

The World Bank. World development indicators | databank: Gdp growth (annual %) (ny.gdp.mktp.kd.zg).

Aggregator	ChartQA	WB-ChartExtract
Mean	84.55	49.72
Huber	86.47	51.39
Median	**86.53**	**51.72**

Table 7: Aggregation method ablation using Llama 4 Maverick, evaluated with RMS_{F1}.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. *Preprint*, arXiv:2203.11171.

Renqiu Xia, Bo Zhang, Hancheng Ye, Xiangchao Yan, Qi Liu, Hongbin Zhou, Zijun Chen, Peng Ye, Min Dou, Botian Shi, Junchi Yan, and Yu Qiao. 2025. Chartx & chartvlm: A versatile benchmark and foundation model for complicated chart reasoning. *Preprint*, arXiv:2402.12185.

Liang Zhang, Anwen Hu, Haiyang Xu, Ming Yan, Yichen Xu, Qin Jin, Ji Zhang, and Fei Huang. 2024. TinyChart: Efficient chart understanding with program-of-thoughts learning and visual token merging. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 1882–1898, Miami, Florida, USA. Association for Computational Linguistics.

A Aggregation Method Ablation

We ablate the cell-wise aggregation function using Llama 4 Maverick, comparing mean, Huber estimator (Huber, 1964), and median (Table 7). Median and Huber perform similarly, with median achieving the best RMS_{F1}, while mean is substantially worse due to outlier sensitivity. We therefore use the median in all experiments.