

# Scaling Unverifiable Rewards: A Case Study on Visual Insights

Shuyu Gan, James Mooney, Pan Hao, Renxiang Wang,  
Mingyi Hong, Qianwen Wang, Dongyeop Kang

University of Minnesota

{gan00067, moone174, pan00342, wan03409, mhong, qianwen, dongyeop}@umn.edu

## Abstract

Large Language Model (LLM) agents can increasingly automate complex reasoning through Test-Time Scaling (TTS), an iterative refinement process guided by reward signals. However, many real-world tasks involve multi-stage pipelines whose final outcomes lack verifiable rewards or sufficient data to train robust reward models, making judge-based refinement prone to error accumulation across stages. We propose **Selective TTS**, a *process-based refinement* framework that scales inference across stages of a multi-agent pipeline, instead of repeatedly refining a single output over time as in prior work. By distributing compute across stages and pruning low-quality branches early using process-specific judges, Selective TTS mitigates the judge drift and stabilizes refinement. Grounded in a data science workflow, we build an end-to-end multi-agent pipeline for generating visually insightful reports from a given dataset, and design a reliable LLM-based judge model that aligns with human experts (Kendall’s  $\tau=0.55$ ) to evaluate them. Our proposed selective TTS then improves insight quality under a fixed compute budget, increasing mean scores from **61.64** (baseline) to **65.86** while reducing variance. We hope our findings serve as the first step toward scaling complex, open-ended tasks with unverifiable rewards like scientific discovery. Our code and generated reports are publicly available <sup>1</sup>

## 1 Introduction

LLM agents increasingly improve performance by allocating more inference-time compute to *branching*, *reflection*, and *verification*, before committing to answers. Methods such as Self-Consistency (Wang et al., 2023), Tree-of-Thoughts (Yao et al., 2023), and evolutionary refinement like AlphaEvolue (Novikov et al., 2025) demonstrate that iterative test-time reasoning can scale solution quality

<sup>1</sup><https://minnesotanlp.github.io/insight-scaling-webpage>

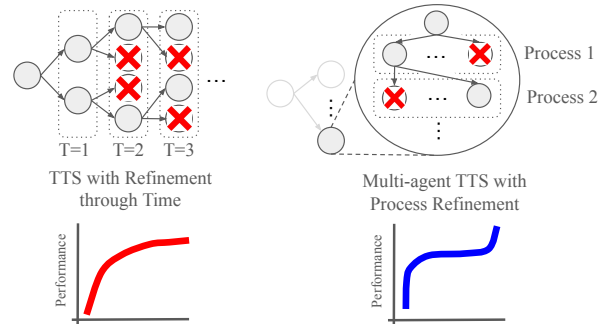


Figure 1: Comparison between traditional TTS through refinement over time (left) (Novikov et al., 2025), and our proposed process refinement in multi-agent pipeline (right). They show different styles of scaling behaviors.

comparably to gains from larger models or additional training (Snell et al., 2024; Liu et al., 2025b).

Many real-world tasks operate as multi-stage pipelines, involving analysis, generation, evaluation, and refinement, where final outcomes are often *unverifiable*, such as evaluating the insightfulness of a chart. These objectives typically lack sufficient data to train robust reward models due to the difficulty and underlying principles of defining the problem, leading practitioners to rely on LLM-as-Judge feedback (Zheng et al., 2023). However, repeated refinement driven by judge scores tends to accumulate early misjudgments, causing traditional time-based scaling to drift away from human-preferred outcomes (Das et al., 2024).

To address these challenges, we propose **Selective Test-Time Scaling (Selective TTS)**, a *process-based, stage-wise* refinement strategy for multi-agent systems. Instead of repeatedly refining a single output through time, Selective TTS allocates compute across pipeline stages and applies stage-specific evaluators to prune low-quality branches early. By confining each judge’s influence to its local stage, this approach reduces error propagation and yields more stable, interpretable scaling behavior under fixed compute (Fig. 1).

Data analysis exemplifies this challenge, as trans-

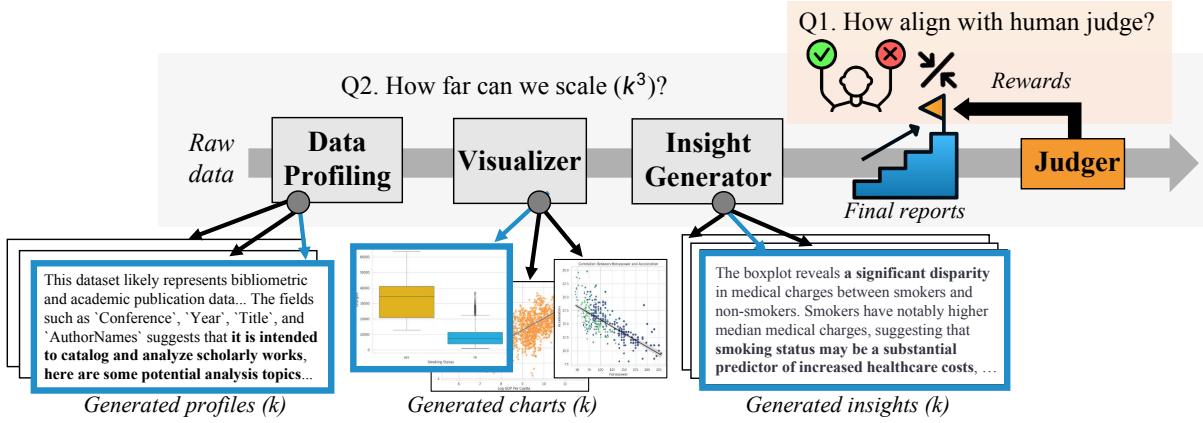


Figure 2: Our multi-agent data analysis pipeline aims to answer two key questions: (RQ1) whether judge-guided scaling can align with human experts, and (RQ2) how much performance can be further scaled within the same compute budget using the proposed selective TTS.

forming raw data into visualizations and insight driven reports requires substantial domain and subjective judgment, making evaluation with a single scalar reward impractical (Wang et al., 2025b). Most existing automated data analysis systems (Google Labs), follow rigid linear workflows that primarily produce surface-level charts.

We adopt this inherently unverifiable setting as a case study to systematically examine how *process-based test-time refinements* can improve end-to-end pipeline quality under a fixed compute budget. Our contributions are threefold:

- We build an end-to-end data analysis pipeline, from raw data to final reports, decomposing the workflow into data profiling, chart generation, insight extraction and verification.
- We design stage-wise *LLM-as-Judges* to evaluate the report quality, and identify a pseudo ground-truth judge that best aligns with human expert judgment (Kendall’s  $\tau=0.55$ ) for subsequent experiments.
- We propose and evaluate *Selective TTS*, a pruning-based compute allocation strategy across pipeline stages, that improves insight quality of reports (+4.2 gain, 39% variance reduction) under the same compute budget.

These contributions reframe this unverifiable insight optimization as a *test-time scaling and allocation* problem, demonstrating a principled recipe to scale outcome quality under unverifiable rewards, while bridging progress in agentic TTS with real-world data science workflows.

## 2 Proposed Methods

We investigate whether unverifiable rewards in visual insight generation can be operationalized and

improved via test-time scaling. To this end, we build a simple multi-agent pipeline that produces insightful reports from raw data, together with a human-aligned LLM-based judge to evaluate report quality (§2.1). Using this judging signal, we then extend our work to see how far we may scale quality under a fixed compute budget (§2.2).

### 2.1 Multi-Agent Data Analysis Pipeline

As shown in Fig. 2, the pipeline takes raw data and produces a final report with curated charts and chart-grounded insights. It comprises four stages: *Data Profiling*, *Visualization*, and *Insight Generation* and *Judge Verification* each implemented as an agent with stage-specific prompts, powered by an LLM or a vision-language model (VLM). The overall design is inspired by the work by (Gan et al., 2025) but simplified for the sake of efficient scaling. Further details on the pipeline design and example stage-wise outputs are provided in Appendix §H.

**(1) Data Profiling.** The *Data Profiling* stage summarizes key metadata of the dataset (shape, schema, inferred types, small samples) and emits a concise *metadata report* with plausible analysis directions. This compact contract guides downstream visualization (valid columns, sensible encoding), reduce hallucinations and routine failures (empty plots), and avoid streaming raw records to the model, improving robustness and efficiency.

**(2) Visualization.** Conditioned on the metadata report, the *Visualization* stage: (i) proposes analysis directions (topic, chart\_type, variables); (ii) compiles executable plotting code; (iii) executes and, if needed, *rectifies* code using error traces; and (iv) filters meaningless figures via a chart judge.

Easy Judge	Moderate Judge	Harsh Judge
<b>Task:</b> objective evaluation using chart-only evidence. <b>Traits:</b> Readability; OnTopic; TrendAlignment. <b>Process:</b> direct observation and scoring. <b>Scoring:</b> integers 0–100 per trait. <b>Output:</b> JSON {scores, evidence, conclusion}.	<b>Task:</b> objective evaluation using chart-only evidence. <b>Traits:</b> Correctness; Specificity; InterpretiveValue. <b>Process:</b> identify chart elements and rate insight clarity. <b>Scoring:</b> integers 0–100 per trait. <b>Output:</b> JSON {scores, evidence, conclusion}.	<b>Task:</b> objective evaluation using chart-only evidence. <b>Traits:</b> Correctness and Factuality; Specificity and Traceability; Insightfulness and Depth; So-what quality. <b>CoT Process:</b> observe chart → decompose insight → map evidence → score → conclude. <b>Scoring:</b> integers 0–100 per trait. <b>Output:</b> JSON {scores, evidence, conclusion}.

Table 1: Prompt snippets for the three judges. The overall evaluation task, scoring scale, and output format are consistent, while the evaluation traits and reasoning process become progressively more demanding. The final evaluation score is computed externally as the mean of the summed trait scores. Full templates are in Appendix §H.3.

As a result, the pipeline yields a vetted set of figures ready for interpretation.

**(3) Insight Generation.** For each vetted chart, the *Insight Generation* stage samples textual insights under specific instructions that emphasize (i) observation fidelity and evidential completeness, (ii) traceability to specific chart elements, (iii) non-triviality and novelty, and (iv) actionability. Each generated insight is evaluated by an LLM-based judge together with its corresponding chart.

**Scaling and Ranking with Judges** Each (chart, insight) pair is summarized into a *final report*. Because each stage produces multiple candidates, the total number of end-to-end report candidates grows combinatorially. Fixing the branching factor of each stage to  $k$  yields up to  $k^3$  possible report candidates. A *Judge* agent then assigns a scalar score  $s \in [0, 100]$  by averaging its per-dimension ratings (e.g., correctness, traceability, nontriviality, actionability). We instantiate three strictness levels—*easy*, *moderate*, and *harsh*—and rank final reports by their scores for downstream selection. Table 1 provides prompt snippets for these judges, designed with support from domain experts and inspired by prior work on insight evaluation and characterization (Law et al., 2020; He et al., 2021; Lian et al., 2025). To improve robustness and reduce stochastic judgment noise, each report is evaluated *three* independent times by the same judge, and the final score is obtained via averaging the three repeated scores. This stabilizes ranking under test-time branching and ensures consistency across evaluation runs.

**Human Alignment of Judges** While judge-based ranking enables scalable selection under combinatorial growth, prompting-based judgment remains inherently imperfect, even when prompts are carefully designed using grounded literature

and domain expertise. Different judges may exhibit systematic biases or preference shifts, making it unclear whether judge-guided improvements genuinely scale to reflect human-perceived quality.

To ensure that judge-guided scaling aligns with human expert preferences, we explicitly validate and select a judge whose rankings are consistent with human judgments. We therefore conduct a human evaluation using pairwise preference rankings over sampled final reports, and measure the agreement between human rankings and each candidate judge. This process allows us to identify a judge that reliably reflects human quality assessments, which is then used as the proxy signal for evaluating report quality under test-time scaling. Experimental details are in §3.1

## 2.2 Scaling Up with Selective Pruning

Having established an end-to-end pipeline for visual insight generation and a human-aligned judging framework in §2.1, we now test how far we can scale insight quality under a fixed compute budget (i.e., finding the best outcome out of a combination of  $k$  profiles,  $k$  charts, and  $k$  reports in Fig. 2).

This section introduces *Selective Test-Time Scaling* (Selective TTS), a process-based pruning strategy that allocates compute more efficiently across multi-agent stages. Unlike naïve test-time scaling that exhaustively expands all branches, Selective TTS prunes low-quality candidates early based on stage-local evaluations, allowing compute to be concentrated on more promising paths (Fig. 3).

### 2.2.1 Stage-Wise Selective TTS

Selective TTS reformulates test-time scaling along the *process* dimension rather than the *temporal* one. Each round of runs proceeds through the same three core stages, data profiling, visualization, and insight generation, but with *stage-local evaluators* that prune weak candidates before passing results

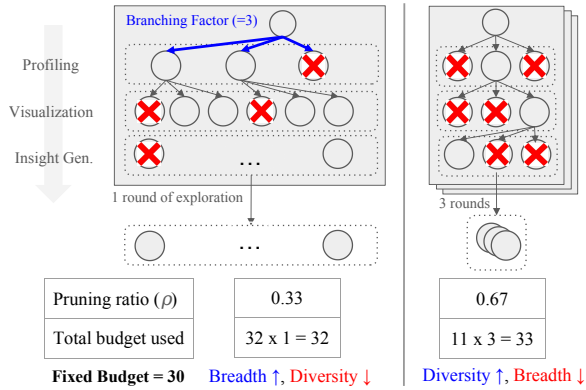


Figure 3: Overview of **Selective Test-Time Scaling** under a fixed compute budget (e.g., 32). Stage-specific evaluators perform **pruning** across the multi-agent pipeline while controlling the **branching** factor at each stage. Higher pruning ratios ( $\rho$ ) promote more rounds of exploration and diversity across runs, while reducing within-round breadth. Details on budget accounting and branching control appear in §2.2.2.

downstream. For instance, the data profiling stage produces multiple metadata reports; the visualization stage expands each report into several visualization specifications (topic, variable, chart type) and render the charts; and the insight generation stage drafts multiple textual insights per chart. After *every* stage, a dedicated LLM-based stage-local evaluator ranks candidates and discards a fixed fraction of low-quality outputs, enabling more compute to be allocated to high-potential branches (Fig. 3).

**Pruning Schedule.** Let  $b_s$  denote the branching factor at stage  $s \in \{\text{data profiling, visualization, insight generation}\}$ . Given a pruning ratio  $\rho$ , we retain

$$n'_s = \max(1, \lceil (1 - \rho) b_s \rceil)$$

candidates to pass forward.<sup>2</sup> Thus  $\rho = 0$  corresponds to the baseline (no pruning, full branching at every stage), while larger  $\rho$  increases selectivity by trimming more aggressively. We apply uniform  $\rho$  across stages for simplicity, though state-specific pruning schedules are possible.

**Stage-local Evaluators.** Each pruning decision is guided by an LLM-based judge tailored to its stage: a *metadata report judge* ranks metadata reports, a *visualization judge* ranks directions of visual charts, and an *insight judge* ranks drafted insights. This process-level allocation reduces the ac-

<sup>2</sup>We enforce  $\max(1, \cdot)$  to avoid pathological collapse at high pruning rates.

cumulation of judging noise across time and yields more stable scaling behavior under fixed budget.

## 2.2.2 Budget Control and Complexity

To ensure fair comparison across pruning ratios, we keep the *total inference compute* constant for all Selective TTS settings. We measure compute by the number of LLM calls (generation, stage-local judging, and final evaluation) aggregated across all agents. Because all configurations share the same model and execution environment, call-based budgeting provides a stable and reproducible proxy for inference cost without requiring token-level accounting, with detailed per-stage call accounting provided in Appendix §E.1. Given a baseline budget  $B$  at  $\rho=0$ , we adjust the number of runs under each pruning ratio  $\rho$  so that the accumulated call count closely matches  $B$ . During execution, we monitor the budget online and incrementally launch additional runs until the target budget is reached (The normalization procedure is provided in Appendix §E.2). This normalization ensures that any performance gains reflect a more effective compute allocation rather than the increased compute, and we further verify in §4.4 that call-based budgeting yields trends consistent with token-level budgets.

Under this budget model, the expected run-level budget complexity can be approximated as

$$\mathbb{E}[B_{\text{run}}(\rho)] \approx \underbrace{\mathcal{O}(p_v n_s'^3)}_{\text{Generation Budget}} + \underbrace{\mathcal{O}(\mathbb{I}[\rho > 0] p_v n_s'^2)}_{\text{Pruning Budget}},$$

where  $n'_s = \max(1, \lceil (1 - \rho) b_s \rceil)$  denotes the number of surviving branches after pruning at stage  $s$ , and  $p_v \in [0, 1]$  is the expected pass rate of the visualization-stage quality filter. Intuitively,  $p_v n'_s$  is the expected number of verified charts among the  $n'_s$  visualization directions. The first term captures the dominant generation and evaluation cost, which scales *cubically* with the effective branching factor due to combinatorial expansion across stages. The second term is the additional pruning overhead, which scales *quadratically* in the retained branch size and is incurred only when pruning is enabled. A full derivation is provided in Appendix §E.1.

## 3 Experimental Setup

We design two complementary experimental setups corresponding to our two research questions: (1) the *small-sized human alignment experiment* to validate our judge selection (§2.1), and (2) the *large-scale scaling experiment* to show effectiveness of selective TTS (§2.2).

Judge	VIS Publication Dataset			Medical Insurance Dataset		
	Kendall’s $\tau$ ( $\uparrow$ )	Spearman’s $\rho$ ( $\uparrow$ )	Kendall’s $W$ ( $\uparrow$ )	Kendall’s $\tau$ ( $\uparrow$ )	Spearman’s $\rho$ ( $\uparrow$ )	Kendall’s $W$ ( $\uparrow$ )
Easy	0.40 $\pm$ 0.24	0.53 $\pm$ 0.24	<b>0.64</b>	0.55 $\pm$ 0.30	0.62 $\pm$ 0.26	0.54
Moderate	0.45 $\pm$ 0.17	0.55 $\pm$ 0.23	0.51	0.40 $\pm$ 0.00	0.55 $\pm$ 0.08	<b>0.65</b>
<b>Harsh</b>	<b>0.55<math>\pm</math>0.30</b>	<b>0.60<math>\pm</math>0.37</b>	0.59	<b>0.65<math>\pm</math>0.30</b>	<b>0.72<math>\pm</math>0.27</b>	0.64

Table 2: Alignment metrics (mean  $\pm$  std over *four* human annotators) between each judge and human annotations on two datasets. Higher values indicate stronger agreement (1 denotes perfect correlation).

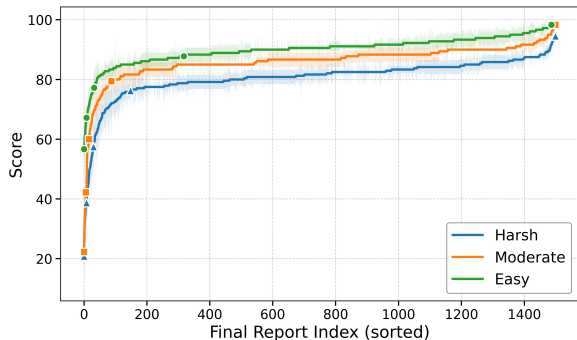


Figure 4: Sorted overall score curves under three judges (easy, moderate, harsh) on VIS Publication dataset. See Appendix §C for the Medical Insurance dataset.

### 3.1 Human Alignment Experiment

We first run the full Selective TTS pipeline under test-time scaling to generate a large set of final reports with varying quality levels. Based on the scores produced by *three* judges with different strictness levels, we then adopt a score-stratified sampling strategy to obtain representative samples across the full quality spectrum, as illustrated by the sorted score curves in Fig. 4. Specifically, reports are ordered by judge scores and sampled at five fixed score quantiles (0%, 25%, ... 100%).

Pairwise human preferences are collected from *four* expert annotators with extensive experience in data science. These annotations are used to construct a consensus human ranking, and alignment between each judge’s ranking and human preferences is evaluated using rank-based correlation metrics, with inter-rater agreement assessed to ensure annotation reliability. Based on this alignment analysis, we identify a pseudo ground-truth judge that best matches human judgment.

For report generation, we use Qwen2.5-VL-32B-Instruct as the generation backbone and GPT-4o as a strong judging backbone to minimize model-induced bias. Each report is evaluated by the same judge three independent times and the final score is obtained by averaging these evaluations. Experiments are conducted on two representative tabular datasets—*VIS Publication* (Isenberg et al., 2017) and *Medical Insurance* (Abdelghany, 2021)—each

yielding approximately 1,500 final reports. These evaluations produce sorted index curves that characterize the score distribution across pruning ratios and serve as the basis for subsequent human-alignment analysis. Full experimental details, annotation protocols, and statistical definitions are provided in Appendix §D.1

### 3.2 Scaling Experiment with Selective TTS

#### Model, Dataset, and Inference Configuration.

All main experiments are conducted on the *VIS Publication Dataset* (Isenberg et al., 2017), and we adopt Qwen2.5-VL-32B-Instruct as the generation backbone for all agents. To obtain more accurate and stable evaluations, we employ GPT-4.1-nano as the judging backbone (stage-local evaluators and final judge) throughout the scaling experiments. To ensure sufficient diversity in test-time scaling, we set the decoding parameters to top- $p = 0.9$ , temperature = 1.0, and a maximum generation length of 1500 tokens. We fix the stage-local branching factor to  $b_s = 5$ , ensuring consistent diversity across all stages. Pruning applies a single ratio  $\rho$  uniformly across stages within a run; we sweep  $\rho \in \{0.2, 0.4, 0.6, 0.8\}$ , while all other settings are held fixed across runs. In addition, we later conduct robustness analyses examining budget selection, cross-model variation, decoding parameter choices, dataset generalization and prompt-controlled variation in base insight length in §4.4

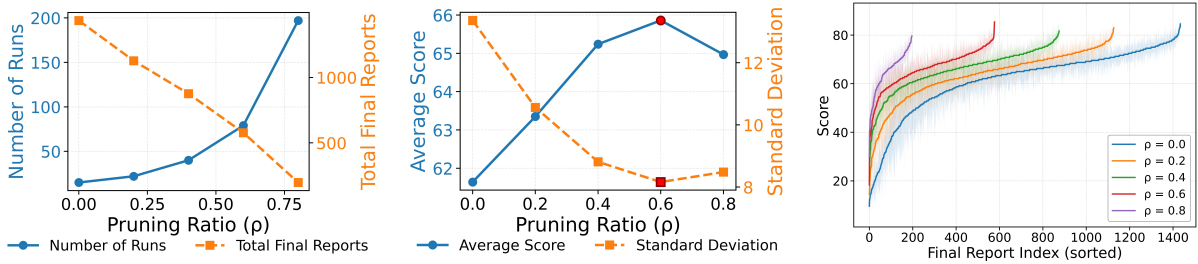
**Baseline Budget.** We establish a reference budget by running the full pipeline with no pruning ( $\rho=0$ ) for 15 independent runs. All Selective TTS conditions are matched to this budget (within a small tolerance) to isolate *allocation* effects from total compute.

## 4 Results

We like to answer the two key research questions:

**RQ1:** Do judge-guided, test-time-scaled reports align with human expert preferences? (§4.1)

**RQ2:** Given alignment, how far can we scale quality under a fixed compute budget? (§4.2–§4.5)



(a) Runs (left) and total final reports (right) vs. pruning ratio  $\rho$  (b) Average score (left) and standard deviation (right) vs. pruning ratio  $\rho$  (c) Sorted score curves under different  $\rho$ .

**Figure 5: Effects of pruning ratio  $\rho$  on compute allocation and quality.** (a) Higher  $\rho$  shifts compute from within-run breadth (fewer reports per run) to cross-run exploration (more runs). (b) The mean score increases with stronger pruning and peaks at  $\rho = 0.6$ , while the standard deviation generally decreases, indicating improved stability in report quality. (c) Increasing  $\rho$  removes the low-quality tail and shifts the score distribution upward.

#### 4.1 Judge Selection via Human Alignment

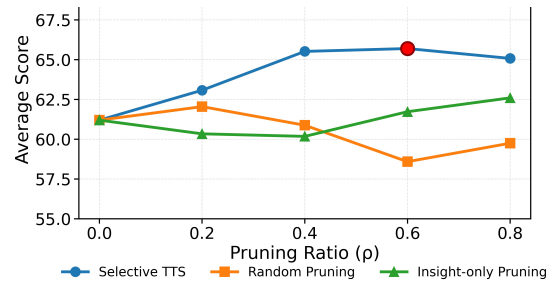
Table 2 shows that the **harsh** judge best aligns with human preferences across both datasets (highest  $\tau$  and  $\rho$ ), with a relatively strong inter-rater agreement reflected by  $W$ . We adopt it as the pseudo ground-truth objective for all subsequent **Selective TTS** experiments. This establishes (i) a modular, auditable pipeline with exposed intermediate artifacts, and (ii) a human-calibrated scalar reward that is sufficiently discriminative to guide stage-wise pruning. We demonstrate that selective, stage-wise pruning improves mean insight quality and reduces variance under a fixed compute budget, providing a practical recipe for scaling visual insight generation when rewards are unavailable.

A representative example of the human preference comparison, shown together with the corresponding judge scores, is provided in Appendix §D.5. Further details on the annotation interface, protocol, and rater calibration procedure can be found in Appendix §D.3 and Appendix §D.4.

#### 4.2 Selective Pruning for Scaling

We test whether Selective TTS improves overall report quality under a fixed compute budget. Figs. 5a- 5c summarize results.

**Compute Allocation under Pruning.** As  $\rho$  increases, the total number of final reports (i.e., chart-insight pairs) decreases substantially (Fig. 5a, right axis and Tab. 4), since weak branches are discarded early. Under the same overall budget, this enables *more* independent runs (left axis), *trading within-run breadth for cross-run exploration*. The total number of reports is not simply  $\text{Runs} \cdot n'_s{}^3$  with  $n'_s = \max(1, \lceil (1-\rho) b_s \rceil)$ , because some branches fail quality gates or execution and are dropped; when  $n'_s = 1$  (e.g.,  $\rho = 0.8$ ), each surviving run



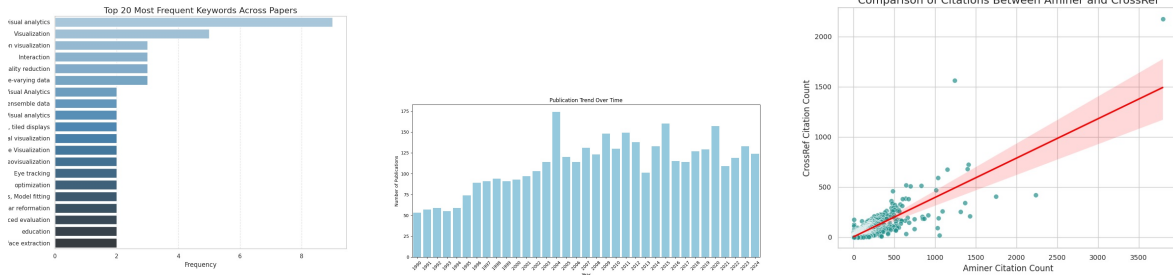
**Figure 6: Ablation.** Average scores comparing Selective TTS with Random Pruning and Final Stage only (Insight-Only) Pruning across varying pruning ratios  $\rho$

deterministically yields one report. *Pruning adds lightweight judging overhead but recovers budget by avoiding low-quality generations downstream*, netting a better allocation under the same total calls.

**Quality and Stability.** Fig. 5b shows that mean scores increase steadily from  $\rho = 0$  to  $\rho = 0.6$  (from 61.64 to 65.86), accompanied by a clear reduction in variance (from 13.36 to 8.16). However, at  $\rho = 0.8$  (*best-of- $N$*  at each stage) the average score drops and the variance rises. This degradation may stem from over-pruning: stage-local evaluators are not always aligned with the final judge, and stronger pruning can amplify these misalignments, potentially removing a larger number of high-quality candidates. Sorted curves (Fig. 5c) show that the lower tail is pruned away and the distribution shifts upwards, while top scores are preserved. Together, these trends indicate that Selective TTS concentrates compute on promising directions, yielding higher mean quality and lower variance under a matched budget. Detailed numbers appear in Appendix §E.3, Table 4.

#### 4.3 Ablation Study

To clarify the role of stage local pruning in Selective TTS, we conduct two ablation studies: replac-



Several keywords, such as "Dimensionality reduction," "Time-varying data," and "Ensemble data," have similar frequencies (3), indicating that these topics are equally important but distinct areas of inquiry. This suggests a diverse set of challenges and methods being explored in handling large and dynamic datasets. Researchers looking to innovate might explore novel combinations of dimensionality reduction techniques with ensemble data analysis, particularly when dealing with temporal data. Conference organizers or professional associations in the field could plan panel discussions or workshops focusing on these interconnected areas to facilitate knowledge sharing and collaboration among experts.

There appears to be a cyclical pattern in the publication trend, with periods of growth followed by stabilization or slight decline. For instance, after the initial growth phase ending in 2004, the number of publications shows stability or minor declines in subsequent years. However, there is a resurgence of growth starting around 2015, leading to another peak in 2020. This cycle may reflect periodic cycles in funding, attention, or major discoveries within the field. Understanding these cycles could help stakeholders anticipate periods of increased activity and plan resources accordingly. For example, research institutions might increase their focus on grant proposals and collaboration efforts ahead of expected growth phases, as seen leading up to the peaks in 2004 and 2020.

The scatter plot reveals a positive correlation between Aminer Citation Count and CrossRef Citation Count, as indicated by the upward-sloping red regression line. This suggests that as the citation count in Aminer increases, there is a corresponding increase in the citation count reported by CrossRef. However, the scatter of data points around the regression line implies variability in this relationship, meaning some papers may have citation counts that deviate significantly from the expected trend. For example, while most points are near the line, there are outliers such as the paper with approximately 3700 Aminer citations but only 400 CrossRef citations. This could indicate discrepancies in how these platforms count citations or differences in their coverage of certain publication types or fields. Given this variability, actors like researchers or publishers might need to consider both citation databases and cross-reference results to ensure a more accurate understanding of a paper's impact.

Figure 7: Generated reports for  $\rho = 0.6$  at the low (32.92), medium (58.3), and high scores (84.2).

ing all stage local evaluators with random sampling, and applying pruning only at the final insight stage. These ablations disentangle whether the gains stem from the full selective mechanism, or from late stage ranking or random exploration. For fair comparison, all experiments are run in a small scale setting with *eight* runs for the Selective TTS baseline, and the LLM call budget is matched across all conditions. Details are reported in Appendix §E.4.

**Random Sampling.** Based on Fig. 6, results show that random pruning yields substantially worse performance, with mean scores decreasing as  $\rho$  increases, despite similar or larger computation. This indicates that unguided pruning harms overall quality, suggesting that stage-local evaluators provide meaningful global benefits by filtering unpromising branches early.

**Final Stage only Pruning.** When pruning is applied only during insight generation (Fig. 6), performance remains largely unchanged at small pruning ratios and shows only modest gains under aggressive pruning, which are substantially smaller than those achieved by full Selective TTS. Even at  $\rho = 0.8$ , where insight only pruning effectively resembles a *best-of-N* selection over final reports, performance still lags behind full Selective TTS. Because earlier stages are not pruned, many low quality branches persist to the final stage, limiting the impact of late stage selection alone.

#### 4.4 Robustness of Scaling

In the following analyses, we examine whether the scaling behavior of Selective TTS is robust

across multiple dimensions to address the following questions:

- 1) **Budget accounting.** Does the observed scaling behavior depend on using LLM calls versus token-level compute budgets?
- 2) **Backbone.** Does Selective TTS generalize across different generator-judger model families?
- 3) **Decoding parameters.** Is scaling behavior sensitive to decoding strategies and generation diversity?
- 4) **Dataset generalization.** Does Selective TTS transfer beyond the VIS Publication dataset to other data domains?
- 5) **Insight length.** Does variation in prompt-controlled base insight length affect the scaling behavior of Selective TTS?

Across all five dimensions, we observe consistent and stable scaling trends: stage-wise pruning reliably improves performance, and the qualitative behavior of Selective TTS is preserved across compute definitions, model backbones, decoding regimes, datasets, and base insight length variations. Detailed experimental setups and qualitative analyses are provided in Appendix §E.5.

#### 4.5 Human Evaluation of Generated Reports

We sample reports with low, medium, and high automatic scores across all pruning ratios and collect pairwise preferences from four annotators.

**Qualitative Evaluation** Within each pruning ratio, higher scoring reports are consistently more focused, evidence grounded, and reliable. For example, at  $\rho = 0.6$  (Fig. 7), the lowest scoring report

Reports	Kendall's $\tau$ ( $\uparrow$ )	Spearman's $\rho$ ( $\uparrow$ )	Kendall's W
Low (24.58 $\pm$ 13.94)	<b>0.70<math>\pm</math>0.10</b>	<b>0.82<math>\pm</math>0.08</b>	<b>0.7875</b>
Medium (54.83 $\pm$ 6.63)	0.30 $\pm$ 0.41	0.38 $\pm$ 0.45	0.2750
High (84.67 $\pm$ 0.49)	-0.15 $\pm$ 0.17	-0.17 $\pm$ 0.19	0.1250

Table 3: Alignment metrics between the harsh judge and human ratings for reports sampled from *low*, *medium*, and *high* scoring across all pruning ratios. The alignment decreases in high-quality reports, as the differences become subtle and indistinguishable and annotators rated these top reports comparably strong.

contains only surface-level keywords, the mid scoring report summarizes overall publication trends, and the highest scoring report explains why certain outlier papers show different citation counts across two citation systems.

Increasing  $\rho$  generally improves overall report quality. Even low scoring reports at  $\rho = 0.8$  are more coherent and analytical than those at smaller  $\rho$  (see Fig. 18 in Appendix §F). High scoring reports become similarly informative while remaining diverse in perspective; for instance,  $\rho = 0.8$  focuses on citation discrepancies and outlier samples, whereas  $\rho = 0.4$  highlights anomalies in paper length (Fig. 19 in Appendix §F).

**Pairwise Quality Ranking** Annotators judged high-quality reports to be comparably strong, as reflected by the low variance in preferences and largely indistinguishable ratings reported in Table 3. This trend is further illustrated in Fig. 17 in Appendix §F, where mean annotator rankings across pruning ratios increasingly converge for the highest-ranked reports, indicating that strong reports become difficult to distinguish as their quality improves. Overall, these results suggest that Selective TTS **raises the quality floor** while simultaneously reducing variability among top-ranked outputs, leading to more consistent and reliable insights. Additional qualitative examples and discussion are provided in Appendix §F.

## 5 Related Work

**Test-Time Scaling in LLM Agents.** Test-Time Scaling (TTS) enhances LLM performance by allocating more inference-time compute (Snell et al., 2024; Liu et al., 2025b). Existing approaches are either *sequential*—performing iterative self-refinement or reasoning loops (Madaan et al., 2023; Gou et al., 2024; Zhu et al., 2025; Muennighoff et al., 2025)—or *parallel*, sampling multiple candidates for reward-based aggregation (e.g., Best-of- $N$  (Sun et al., 2024), tree search (Yao et al., 2023)).

Recent studies extend TTS to multi-agent or interactive settings, redistributing compute across reasoning, planning, and verification modules (Zhu et al., 2025; Wang et al., 2025a; Yang et al., 2025; Jin et al., 2025). However, these methods mainly emphasize *temporal refinement* within a single reasoning loop. Our work instead formulates a *stage-wise* TTS that reallocates compute across agents, mitigating judge error propagation and enabling interpretable scaling under fixed budgets.

**Data Science Agents and Visual Insight Generation.** LLM-based agents increasingly automate data-centric workflows like exploratory analysis, code generation, and visualization. Early systems like Google’s *Data Science Agent* (Google Labs), *DS-Agent* (Guo et al., 2024), and *DataInterpreter* (Hong et al., 2024) demonstrated end-to-end automation but remained template-driven and descriptive. Later variants extend to domains such as genomics (Liu et al., 2024), ML benchmarking (Liu et al., 2025a), and healthcare (Merrill et al., 2024).

Recent work further explores *visual insight generation*, where LLMs interpret and contextualize charts (Wang et al., 2025b; Zhao et al., 2025b,a), informed by studies on human insight evaluation (Law et al., 2020; He et al., 2021; Lian et al., 2025). However, most systems remain single-pass without inference-time optimization or stage coordination. We instead adopt this task as a *testbed for selective, stage-wise TTS*, enabling structured multi-agent reasoning and adaptive compute allocation across the pipeline.

## 6 Conclusion

In this work, we reframed unverifiable insight generation as a *test-time scaling* problem and proposed *Selective TTS*, a process-based, stage-wise pruning strategy guided by lightweight LLM-based evaluators. Building on an end-to-end multi-agent pipeline for chart-grounded insight generation, we introduced judge selection via human alignment and formulated compute usage in terms of LLM calls. Our experiments demonstrated that selective pruning under a fixed budget progressively reduces variance, improves average final report quality, and enables broader exploration across runs without requiring additional resources. This study closes a gap between generic reasoning agents and data-centric decision support, showing how LLM-as-Judgers can provide a principled mechanism for scaling unverifiable reasoning.

## 7 Limitations

Our study has several limitations. First, we evaluated only five datasets, which may not fully capture the diversity of real-world data science tasks. Second, although we explored larger compute budgets and an alternative dataset, the overall compute scale and dataset coverage remain moderate.

Future work could address these limitations in several directions. In particular, expanding dataset coverage and model diversity would enable a more comprehensive evaluation. It would also be valuable to explore a wider range of parameter configurations, such as varying the branching factor and pruning ratio, to better understand their effects on selective scaling behavior. Beyond visual insight generation, an important direction is to extend the proposed Selective TTS framework to other unverifiable generative tasks, such as story generation and video generation, where similar challenges of scalability and evaluation arise.

**Potential risks.** As our framework automates unverifiable insight generation, there remains a risk of producing biased or misleading conclusions if model judgments are misaligned with human reasoning. We therefore emphasize that such systems should be used to assist, rather than replace, human analysts, with appropriate calibration, transparency, and human oversight.

## 8 Ethical Considerations

The proposed *SelectTTS* framework does not involve the collection of sensitive user data, nor does it rely on any personally identifiable information. All datasets used are publicly available and contain only aggregated, non-sensitive tabular records. Human annotations were conducted voluntarily by researchers with prior experience in data visualization, without financial incentives, and no personally identifying information was collected.

The goal of this study is methodological, aiming to understand how inference-time compute allocation affects reasoning quality, rather than to deploy autonomous data science systems. Nevertheless, automated insight generation carries potential societal and epistemic risks, including over-reliance on synthetic or unverifiable insights, amplification of bias from training data, and misinterpretation of automatically generated visual explanations.

We encourage responsible deployment by maintaining human oversight in critical decision-making

contexts, transparently reporting model provenance and parameters, and discouraging the use of automatically generated insights for policy or medical decision-making without expert verification.

Finally, all experiments were conducted under controlled research environments, and no additional environmental or privacy risks were introduced beyond standard LLM inference workloads.

## References

- Mosap Abdelghany. 2021. [Medical insurance cost dataset](#). Accessed: 2025-02-11.
- Usama Buttar. 2023. [World happiness report \(2005–present\)](#). Accessed: 2025-02-11.
- Debarati Das, Karin De Langis, Anna Martin-Boyle, Jaehyung Kim, Minhwa Lee, Zae Myung Kim, Shirley Anugrah Hayati, Risako Owan, Bin Hu, Ritik Parkar, Ryan Koo, Jonginn Park, Aahan Tyagi, Libby Ferland, Sanjali Roy, Vincent Liu, and Dongyeop Kang. 2024. [Under the surface: Tracking the artifactuality of llm-generated data](#). *Preprint*, arXiv:2401.14698.
- Minahil Fatima. 2023. [Performance trends in education](#). Accessed: 2025-02-11.
- Shuyu Gan, Renxiang Wang, James Mooney, and Dongyeop Kang. 2025. [A2p-vis: an analyzer-to-presenter agentic pipeline for visual insights generation and reporting](#). *Preprint*, arXiv:2512.22101.
- Google Labs. Data science agent. <https://labs.google.com/code/dsa>. Accessed: 2025-08-18.
- Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. 2024. [Critic: Large language models can self-correct with tool-interactive critiquing](#). *Preprint*, arXiv:2305.11738.
- Siyuan Guo, Cheng Deng, Ying Wen, Hechang Chen, Yi Chang, and Jun Wang. 2024. [DS-agent: Automated data science by empowering large language models with case-based reasoning](#). In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 16813–16848. PMLR.
- Chen He, Luana Micallef, Liye He, Gopal Peddinti, Tero Aittokallio, and Giulio Jacucci. 2021. [Characterizing the quality of insight by interactions: A case study](#). *IEEE Transactions on Visualization and Computer Graphics*, 27(8):3410–3424.
- Sirui Hong, Yizhang Lin, Bang Liu, Bangbang Liu, Binhao Wu, Ceyao Zhang, Chenxing Wei, Danyang Li, Jiaqi Chen, Jiayi Zhang, Jinlin Wang, Li Zhang, Lingyao Zhang, Min Yang, Mingchen Zhuge, Taicheng Guo, Tuo Zhou, Wei Tao, Xiangru Tang, and 8 others. 2024. [Data interpreter: An LLM agent for data science](#). *Preprint*, arXiv:2402.18679.

- Petra Isenberg, Florian Heimerl, Steffen Koch, Tobias Isenberg, Panpan Xu, Chad Stolper, Michael Sedlmair, Jian Chen, Torsten Möller, and John Stasko. 2017. [vispubdata.org: A metadata collection about IEEE visualization \(VIS\) publications](https://vispubdata.org/). *IEEE Transactions on Visualization and Computer Graphics*, 23(9):2199–2206.
- Can Jin, Hongwu Peng, Qixin Zhang, Yujin Tang, Dimitris N. Metaxas, and Tong Che. 2025. [Two heads are better than one: Test-time scaling of multi-agent collaborative reasoning](#). *Preprint*, arXiv:2504.09772.
- Akshay Dattatray Khare. 2020. [Diabetes dataset](#). Accessed: 2025-02-11.
- Po-Ming Law, Alex Endert, and John Stasko. 2020. [What are data insights to professional visualization users?](#) *Preprint*, arXiv:2008.13057.
- Yijie Lian, Jianing Hao, Wei Zeng, and Qiong Luo. 2025. [A survey of visual insight mining: Connecting data and insights via visualization](#). *Visual Informatics*, page 100271.
- Haokun Liu, Sicong Huang, Jingyu Hu, Yangqiaoyu Zhou, and Chenhao Tan. 2025a. [Hypobench: Towards systematic and principled benchmarking for hypothesis generation](#). *Preprint*, arXiv:2504.11524.
- Haoyang Liu, Shuyu Chen, Ye Zhang, and Haohan Wang. 2024. [Genotex: An LLM agent benchmark for automated gene expression data analysis](#). *Preprint*, arXiv:2406.15341.
- Runze Liu, Junqi Gao, Jian Zhao, Kaiyan Zhang, Xiu Li, Biqing Qi, Wanli Ouyang, and Bowen Zhou. 2025b. [Can 1b llm surpass 405b llm? rethinking compute-optimal test-time scaling](#). *Preprint*, arXiv:2502.06703.
- Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegrefe, Uri Alon, Nouha Dziri, Shrimai Prabhunoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. 2023. [Self-refine: Iterative refinement with self-feedback](#). *Preprint*, arXiv:2303.17651.
- Mike A. Merrill, Akshay Paruchuri, Naghmeh Rezaei, Geza Kovacs, Javier Perez, Yun Liu, Erik Schenck, Nova Hammerquist, Jake Sunshine, Shyam Taylor, Kumar Ayush, Hao-Wei Su, Qian He, Cory Y. McLean, Mark Malhotra, Shwetak Patel, Jiening Zhan, Tim Althoff, Daniel McDuff, and Xin Liu. 2024. [Transforming wearable data into health insights using large language model agents](#). *Preprint*, arXiv:2406.06464.
- Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. 2025. [s1: Simple test-time scaling](#). *Preprint*, arXiv:2501.19393.
- Alexander Novikov, Ngô Vũ, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, Borislav Kozlovskii, Francisco J. R. Ruiz, Abbas Mehrabian, M. Pawan Kumar, Abigail See, Swarat Chaudhuri, George Holland, Alex Davies, Sebastian Nowozin, Pushmeet Kohli, and Matej Balog. 2025. [Alphaevolve: A coding agent for scientific and algorithmic discovery](#). *Preprint*, arXiv:2506.13131.
- Ayesha Siddiq. 2022. [Customer shopping behavior dataset](#). Accessed: 2025-02-11.
- 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.
- Hanshi Sun, Momin Haider, Ruiqi Zhang, Huitao Yang, Jiahao Qiu, Ming Yin, Mengdi Wang, Peter Bartlett, and Andrea Zanette. 2024. [Fast best-of-n decoding via speculative rejection](#). *Preprint*, arXiv:2410.20290.
- Fali Wang, Hui Liu, Zhenwei Dai, Jingying Zeng, Zhiwei Zhang, Zongyu Wu, Chen Luo, Zhen Li, Xianfeng Tang, Qi He, and Suhang Wang. 2025a. [Agenttts: Large language model agent for test-time compute-optimal scaling strategy in complex tasks](#). *Preprint*, arXiv:2508.00890.
- Fen Wang, Bomiao Wang, Xueli Shu, Zhen Liu, Zekai Shao, Chao Liu, and Siming Chen. 2025b. [Chartinsighter: An approach for mitigating hallucination in time-series chart summary generation with a benchmark dataset](#). *Preprint*, arXiv:2501.09349.
- 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.
- Yan Yang, Dongxu Li, Yutong Dai, Yuhao Yang, Ziyang Luo, Zirui Zhao, Zhiyuan Hu, Junzhe Huang, Amrita Saha, Zeyuan Chen, Ran Xu, Liyuan Pan, Caiming Xiong, and Junnan Li. 2025. [Gta1: Gui test-time scaling agent](#). *Preprint*, arXiv:2507.05791.
- Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. [Tree of thoughts: Deliberate problem solving with large language models](#). *Preprint*, arXiv:2305.10601.
- Yuheng Zhao, Junjie Wang, Linbing Xiang, Xiaowen Zhang, Zifei Guo, Cagatay Turkay, Yu Zhang, and Siming Chen. 2025a. [Lightva: Lightweight visual analytics with llm agent-based task planning and execution](#). *IEEE Transactions on Visualization and Computer Graphics*, 31(9):6162–6177.
- Yuheng Zhao, Yixing Zhang, Yu Zhang, Xinyi Zhao, Junjie Wang, Zekai Shao, Cagatay Turkay, and Siming Chen. 2025b. [Leva: Using large language](#)

models to enhance visual analytics. *IEEE Transactions on Visualization and Computer Graphics*, 31(3):1830–1847.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. [Judging llm-as-a-judge with mt-bench and chatbot arena](#). *Preprint*, arXiv:2306.05685.

King Zhu, Hanhao Li, Siwei Wu, Tianshun Xing, Dehua Ma, Xiangru Tang, Minghao Liu, Jian Yang, Jiaheng Liu, Yuchen Eleanor Jiang, Changwang Zhang, Chenghua Lin, Jun Wang, Ge Zhang, and Wangchunshu Zhou. 2025. [Scaling test-time compute for llm agents](#). *Preprint*, arXiv:2506.12928.

## A Use of LLMs

We used ChatGPT(chatgpt.com) to generate structured sentences as placeholders then paraphrased and refined in our own words.

## B Datasets

We conduct all experiments on six public datasets designed for visualization-based reasoning and data analysis tasks: *VIS Publication* (Isenberg et al., 2017), *Medical Insurance* (Abdelghany, 2021), *Diabetes* (Khare, 2020), *Education Performance* (Fatima, 2023), *Customer Shopping Behavior* (Siddiq, 2022), and the *World Happiness Report* (Buttar, 2023). All datasets are publicly available and released under open research licenses (e.g., CC-BY 4.0). They contain structured tabular data paired with visualizations, enabling evaluation of chart-grounded insight generation.

**VIS Publication.** The VIS Publication dataset is curated from data tables used in published IEEE VIS conference papers. Each sample consists of a structured data table with multiple numerical and categorical attributes, from which candidate visualizations and corresponding descriptive insights can be derived. The dataset covers a diverse range of visualization types (e.g., bar, line, scatter, stacked area) and real-world analytic topics such as demographics, performance metrics, and experimental results.

**Medical Insurance.** The Medical Insurance dataset contains synthetic records generated from an open statistical dataset describing personal attributes (e.g., age, BMI, smoking status, number of children, and insurance charges). It is widely used in data-science education and benchmarking for visualization and regression analysis.

**Diabetes.** The Diabetes dataset contains clinical measurement records collected from patient examinations. Each row includes numerical health indicators such as glucose concentration, blood pressure, skin thickness, insulin level, BMI, and pedigree function, alongside a binary attribute indicating diabetes status. The dataset consists of structured samples combining numerical and binary categorical variables.

**Education Performance.** The Education dataset provides student-level records containing demographic descriptors, study behavior variables, and academic performance measurements. Attributes include gender, parental education, study time, attendance, and exam scores across multiple subjects, represented through both categorical and numerical fields.

**Customer Shopping Behavior.** The Shopping Behavior dataset consists of customer transaction entries from a retail setting. Each record includes product category, unit price, purchase quantity, customer segment, and transaction date. The dataset integrates numerical purchase information with categorical descriptors of customers and products.

**World Happiness Report.** The Happiness dataset aggregates annual country-level indicators reported in the World Happiness Report. Each entry describes socioeconomic and well-being metrics such as GDP per capita, social support, healthy life expectancy, freedom, generosity, and corruption perception. The dataset is composed of numerical indicators organized by country and year.

**Usage and Licensing.** Both datasets are used solely for academic research. No personally identifiable information is included in any record. All visualizations and insights used in our experiments were automatically generated by our pipeline or verified through human calibration.

## C Additional Results on Medical Insurance Dataset

**Observation from Scaling Graph.** Fig. 8 shows the sorted overall score curves on the *Medical Insurance* dataset under the three judgers (**easy**, **moderate**, and **harsh**).

## D Human Annotation

To assess how well the automatic judger aligns with human reasoning, we conducted a structured

Setting	Runs	Total Final Reports	Score (Avg. $\pm$ Std.)	Gen. Budget	Prune Budget
Baseline ( $\rho = 0$ )	15	1435	61.64 $\pm$ 13.36	2567	0
$\rho = 0.2$	22	1128	63.35 $\pm$ 10.56	2240	384
$\rho = 0.4$	40	876	65.24 $\pm$ 8.81	2117	439
$\rho = 0.6$	79	578	<b>65.86 <math>\pm</math> 8.16</b>	2032	522
$\rho = 0.8$	197	197	64.97 $\pm$ 8.48	1970	591

Table 4: Main results under different pruning ratios  $\rho$  on the VIS dataset. Selective TTS progressively reduces the number of final reports while improving average quality under a comparable LLM-call budget.

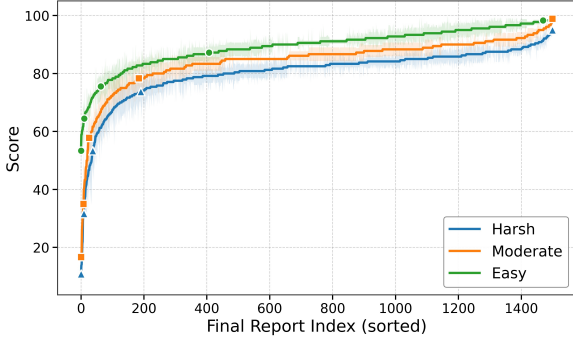


Figure 8: Sorted overall score curves under three judges (easy, moderate, harsh) on Medical Insurance dataset.

human annotation study. This section outlines the overall evaluation framework, including (1) the experimental setup and report generation configuration, (2) the rank-based correlation metrics used to quantify alignment, (3) the annotation interface and protocol for pairwise comparison, (4) the calibration procedures ensuring annotation consistency, and (5) illustrative examples comparing human and model preferences.

### D.1 Experimental Setup and Report Generation Configuration

**Sorted Index Curves.** We make use of the Sorted Index Curves to sample at regular intervals for each judge. This is because, in contrast to refinement based TTS methods (Madaan et al., 2023), we may parallelize the production of each *final report* separately given there are no dependencies between the reports. So, rather than induce a time dimension through arbitrary sampling (which would allow a more direct comparison to TTS with refinement), we instead present the results of many reports sorted in order of the judge scores. We then vertically stratify-sample reports at the 0%, 25%, 50%, 75%, and 100% score quantiles for each judge. Pairwise comparisons among these five quantile-selected reports yield  $3 \times \binom{5}{2} = 30$  representative evaluation points for human preference assessment. For instance, Fig. 4 in §3.1 shows an example sorted score curves by three judges in

§2.1 and five vertical quantiles chosen for human alignment evaluation.

### Human Preference Annotation and Agreement.

The sample points are used to collect pairwise preferences from *four* annotators who have extensive experience in data science and visualization. From these annotations, we first form a consensus human ranking  $\hat{r}$  and evaluate the alignment between each judge’s ranking  $r_j$  and  $\hat{r}$  using Kendall’s  $\tau$  and Spearman’s  $\rho$ . To ensure that the consensus  $\hat{r}$  is itself reliable, we also compute Kendall’s coefficient of concordance  $W$  to assess the *inter-rater agreement* among annotators. A sufficiently high  $W$  indicates that human preferences are stable and do not exhibit severe internal misalignment. Definitions of all three metrics are provided in the §D.2. We then identify the pseudo ground-truth judge  $\tilde{j}$  by selecting the judge with the strongest agreement with human rankings:

$$\tilde{j} = \arg \max_j (\tau_j + \rho_j).$$

The resulting pseudo ground-truth corresponds to the judge most consistent with human judgment (our alignment evaluation are described in §4.1).

### D.2 Kendall’s $\tau$ , Spearman’s $\rho$ , and Kendall’s $W$

To quantify the rank-based alignment between the model-judged ordering  $r_J$  of sampled final reports and the consensus human ranking  $\hat{r}$ , we employ three classical non-parametric statistics: **Kendall’s  $\tau$** , **Spearman’s  $\rho$** , and **Kendall’s  $W$** .

**Kendall’s  $\tau$ .** Given two rankings  $r_J$  and  $\hat{r}$  over  $n$  items, Kendall’s  $\tau$  evaluates their agreement by comparing all  $\frac{1}{2}n(n-1)$  item pairs:

$$\tau_J = \frac{C - D}{\frac{1}{2}n(n-1)},$$

where  $C$  and  $D$  are the numbers of *concordant* and *discordant* pairs, respectively. A pair  $(i, j)$  is

concordant when the two rankings order  $i$  and  $j$  in the same direction:

$$(r_J(i) - r_J(j))(\hat{r}(i) - \hat{r}(j)) > 0.$$

$\tau_J$  ranges from  $-1$  (perfect inversion) to  $1$  (perfect agreement), with  $0$  indicating no systematic correlation. Because it focuses solely on pairwise orderings,  $\tau_J$  is relatively insensitive to large but localized rank shifts.

**Spearman’s  $\rho$ .** Spearman’s rank correlation coefficient instead measures global monotonic agreement by comparing the squared distances between the assigned ranks:

$$\rho_J = 1 - \frac{6 \sum_{i=1}^n (r_J(i) - \hat{r}(i))^2}{n(n^2 - 1)}.$$

Equivalently, it is the Pearson correlation computed over the rank vectors themselves. Like  $\tau_J$ , its value lies in  $[-1, 1]$ . Compared to Kendall’s  $\tau$ , Spearman’s  $\rho$  penalizes large rank deviations more heavily and therefore captures broader structural differences between the two orderings.

**Kendall’s  $W$ .** While  $\tau$  and  $\rho$  describe pairwise agreement between two rankings, Kendall’s coefficient of concordance  $W$  measures the *overall* consistency across  $m$  annotators ranking the same  $n$  items. Let  $R_{.i}$  denote the sum of ranks assigned to item  $i$ , and let  $\bar{R}$  be their mean. Kendall’s  $W$  is defined as:

$$W = \frac{12 \sum_{i=1}^n (R_{.i} - \bar{R})^2}{m^2(n^3 - n)}.$$

$W$  ranges from  $0$  (no agreement beyond chance) to  $1$  (perfect concordance among annotators). Higher values indicate that human annotators produce closely aligned rankings, while lower values reveal substantial disagreement. In our setting,  $W$  establishes the intrinsic consistency of human judgments themselves, contextualizing how well each Judge aligns with human preferences.

**Interpretation.** Kendall’s  $\tau$  emphasizes fine-grained pairwise ordering consistency; Spearman’s  $\rho$  captures global monotonic structure and penalizes large rank deviations; Kendall’s  $W$  summarizes multi-annotator agreement. Together, these metrics offer complementary perspectives on how closely a Judge’s ranking  $r_J$  aligns with the human consensus  $\hat{r}$  and how much inherent variability exists within human annotations.

### D.3 Annotation Interface and Protocol

Fig. 9 illustrates the human annotation interface used in our study. The interface consists of two main sections: the **upper panel** (Fig. 9a) and the **lower panel** (Fig. 9b).

**Interface Layout.** The left sidebar (shared across both panels) displays annotator metadata, including the *Annotator ID*, the selected *Dataset* (e.g., *VIS\_phase\_2*), and the corresponding *Annotation Sequence*. Each annotation session sequentially presents a set of **paired final reports**, where each pair consists of two chart–insight final reports to be compared.

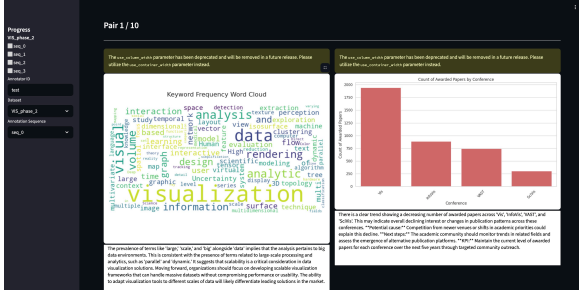
The **upper panel** shows the visual comparison interface: two reports (left and right) are displayed side by side, each containing one visualization and its associated insight text. For a sequence of length  $n$ , all possible  $\binom{n}{2}$  pairs are constructed for pairwise comparison, with both the pair order and the left–right positioning randomized to prevent bias.

The **lower panel** presents the interactive rating sliders used for detailed comparison along multiple rubric dimensions: *Trustworthy / Plausible*, *Complex*, *Unexpectedness*, *Actionability*, *Domain Value*, *Effectiveness*, and *Expressiveness*. Note that, for illustration purposes, not all dimensions are shown in the figure due to layout constraints. Raters are asked to assess which report performs better on each dimension before making an overall decision. This multi-dimensional evaluation encourages more deliberate reasoning, leading to more consistent and reliable judgments.

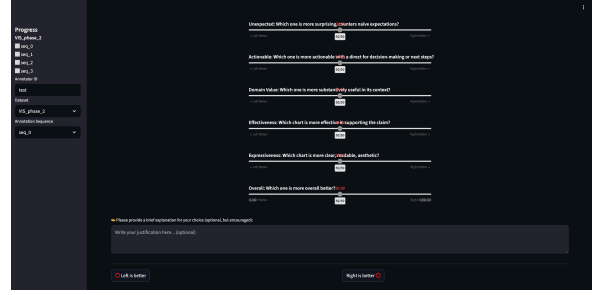
**Annotation Protocol.** Annotators are also encouraged to provide a short textual justification (“*Explain your choice*”) to record their reasoning process. Finally, they indicate the overall better report (*Left is better* or *Right is better*). Across all  $\binom{n}{2}$  comparisons, the number of “wins” for each report is aggregated to obtain a global ranking within the sequence. All results are automatically logged and synchronized to our central server for aggregation and further analysis.

### D.4 Rater Calibration and Adjudication

Before the formal annotation process, all raters underwent a brief **calibration phase** to ensure a consistent understanding of the task and rubric. Each rater first reviewed the corresponding dataset and several representative examples to familiarize themselves with the data characteristics.



(a) Upper panel of the interface displaying two chart–insight reports for side-by-side comparison.



(b) Lower panel showing rubric sliders and overall selection interface.

Figure 9: Human annotation interface used for pairwise comparison of visual–insight reports. The interface consists of an upper comparison panel and a lower evaluation panel.

All annotators were *graduate students with prior experience in data visualization and analytic reasoning*, ensuring familiarity with interpreting charts and evaluating insights.

For each dataset, a designated **pilot annotation sequence** was jointly reviewed by all raters. During this session, raters discussed their interpretations of each rubric dimension and aligned their criteria for scoring and pairwise comparison. This collaborative calibration helped reduce ambiguity and improve inter-rater reliability in the subsequent individual annotations.

After formal annotation, all results were checked for potential anomalies such as inconsistent ranking cycles or missing comparisons. Whenever such irregularities were detected, the affected rater was asked to re-annotate the corresponding sequence to ensure validity and consistency across the final results.

## D.5 Example of Human–Judge Preference Comparison

Fig. 10 illustrates one representative pairwise comparison from the human annotation interface, showing two visual–insight reports displayed side by side along with their corresponding judge scores and human annotator preferences. Each annotator independently selected which report they considered better overall, and the aggregated preferences are shown for illustration. Note that in the actual annotation process, the judge scores were *not visible* to annotators; they were only added here for post-hoc comparison and visualization purposes.

## E Supplementary Analysis and Results for Experiments

This section provides extended analyses and supplementary experiments that complement the main

findings presented in the paper. We begin by deriving the theoretical compute complexity of a single pipeline run and elucidating how pruning affects the overall budget (§E.1). We then describe the budget–matching procedure used in our experiments to ensure fair comparisons across pruning ratios  $\rho$  (§E.2). Following these methodological foundations, we report detailed quantitative results for the main experiments (§E.3), including run counts, final report statistics, and score distributions under matched compute conditions. We further present comprehensive ablation studies (§E.4) that examine the contribution of each stage’s pruning strategy and compare Selective TTS to several simplified baselines. Finally, we provide robustness analyses across budget formulations, backbone models, and decoding configurations (§E.5), offering a broader evaluation of the consistency and generality of Selective TTS.

## E.1 Derivation of Budget Complexity Formula

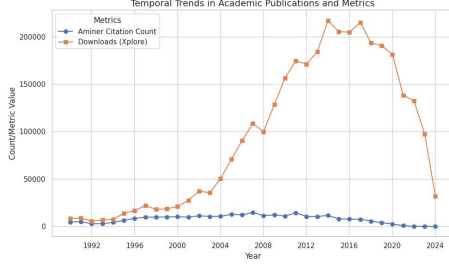
We derive the approximate compute complexity of a single pipeline run under pruning ratio  $\rho$ . For each stage  $s \in \{\text{data profiling, visualization, insight generation}\}$ , let  $b_s$  denote its branching factor, and let

$$n'_s = \max(1, \lceil (1 - \rho)b_s \rceil)$$

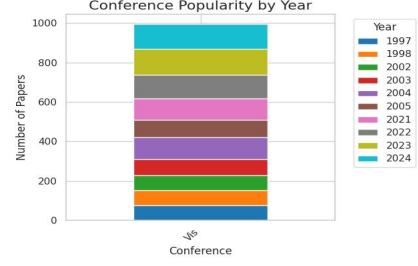
denote the number of retained candidates after pruning at stage  $s$ .

**Budget Accounting.** For each run, the budget is computed as follows:

- *Profiling*:  $+b_s$  call for metadata reports generation, and  $+1$  for pruning (if applied).
- *Visualization*: for each metadata report (profile),  $+1$  for visualization direction generation,



The dramatic decline in academic paper downloads (Xplore) starting around 2020 and continuing to 2024 suggests a significant shift in how researchers access scholarly content. This trend may reflect a growing preference for alternative platforms or open-access resources, possibly due to increased awareness of open science initiatives or changes in institutional policies promoting open repositories. Given that citations (Aminer) have remained relatively stable over the same period, this divergence implies that while researchers continue to cite papers as before, direct access through Xplore has diminished. Academic institutions and publishers should consider investing in user-friendly open-access platforms to maintain accessibility. A potential action could involve launching or optimizing open-access databases, with a goal to see downloads stabilize or rise by 2026, matching pre-2020 levels.



The popularity of conferences, as measured by the number of papers submitted, has shown a consistent decline over time, starting from 1997 to 2024. This trend suggests a possible shift in research focus or declining interest in this particular conference series over nearly three decades. For decision-makers such as conference organizers or academic bodies responsible for these events, investing resources into understanding and reversing this decline could be critical. A potential next step would be to analyze the demographic of attendees or paper submissions for the year 2023, to determine if there is a notable drop-off among specific groups of researchers (e.g., early-career vs. senior researchers). If confirmed, initiatives to attract new participants (e.g., targeted outreach programs or collaboration with industry partners) could be prioritized within the next 12 months to stabilize or increase conference participation.

**Annotation 1: Left one is better**

**Annotation 2: Left one is better**

**Annotation 3: Left one is better**

**Annotation 4: Right one is better**

Figure 10: Representative example of a human-judger preference comparison.

and +1 call for pruning (if applied).

- **Chart Rendering:** +2 calls per visualization direction (code generation + chart quality verification).
- **Insight generation:** For each verified chart, +1 for insight drafting, +1 for pruning (if applied).
- **Judge Verification:** +1 call per chart-insight pair (final report) for quality evaluation

Based on this budget accounting, we can derive the per-run budget complexity.

**Exact Budget Decomposition.** The total compute budget of a single run can be decomposed by enumerating all major LLM calls performed across stages. Specifically, the run-level budget can be written as

$$B_{\text{run}}(\rho) = b_s + \mathbb{I}[\rho > 0] + \sum_{i=1}^{n'_s} (1 + \mathbb{I}[\rho > 0] + 2n'_s + |V_i| + |V_i|\mathbb{I}[\rho > 0] + |V_i|n'_s).$$

where  $|V_i|$  denotes the number of visualization candidates generated from the  $i$ -th retained meta-report that pass chart verification. By construction,  $|V_i| \in \{0, 1, \dots, n'_s\}$  and may vary across different branches.

The terms correspond to the following operations:

- $b_s + \mathbb{I}[\rho > 0]$ : metadata report generation ( $b_s$  LLM calls), with optional pruning;

- $1 + \mathbb{I}[\rho > 0]$ : for each metadata report, visualization direction generation (1 call) and optional pruning;
- $2n'_s$ : for each retained visualization direction, chart code generation (1 call) followed by chart quality verification (1 call);
- $|V_i|$ : for each verified chart, insight generation (1 call);
- $|V_i|\mathbb{I}[\rho > 0]$ : optional pruning applied to generated insights;
- $|V_i|n'_s$ : for each verified chart,  $n'_s$  candidate reports are retained and evaluated by the stage-specific judger.

**Expectation-Based Simplification.** Since  $|V_i|$  varies across branches, we analyze the expected budget. Let  $p_v \in [0, 1]$  denote the *chart verification pass probability* for an individual visualization candidate within a branch (i.e., the probability that a generated chart passes the verification check and is kept for downstream insight generation). Under this formulation,  $|V_i|$  denotes the number of verified charts among the  $n'_s$  attempted visualizations generated from the  $i$ -th retained meta-report. Since each chart passes verification independently with probability  $p_v$ , we model  $|V_i|$  as a binomial random variable with  $n'_s$  trials and success probability  $p_v$ , which implies  $\mathbb{E}[|V_i|] = p_v n'_s$ . Taking expectation

over all branches yields

$$\begin{aligned}\mathbb{E}[B_{\text{run}}(\rho)] &= b_s + \mathbb{I}[\rho > 0] + \sum_{i=1}^{n'_s} \mathbb{E}\left[1 + \mathbb{I}[\rho > 0] + 2n'_s + |V_i| + |V_i|\mathbb{I}[\rho > 0] + |V_i|n'_s\right] \\ &= b_s + \mathbb{I}[\rho > 0] + n'_s(1 + \mathbb{I}[\rho > 0] + 2n'_s + \mathbb{E}[|V_i|] + \mathbb{E}[|V_i|\mathbb{I}[\rho > 0]] + \mathbb{E}[|V_i|]n'_s) \\ &= b_s + \mathbb{I}[\rho > 0] + n'_s(1 + \mathbb{I}[\rho > 0] + 2n'_s + p_v n'_s + p_v n'_s \mathbb{I}[\rho > 0] + p_v n'_s{}^2).\end{aligned}$$

Expanding and regrouping terms by order, we obtain

$$\mathbb{E}[B_{\text{run}}(\rho)] = \underbrace{\mathcal{O}(b_s + n'_s + (2 + p_v)n'_s{}^2 + p_v n'_s{}^3)}_{\text{Expected Generation Budget}} + \underbrace{\mathcal{O}(\mathbb{I}[\rho > 0](1 + n'_s + p_v n'_s{}^2))}_{\text{Expected Pruning Overhead}}.$$

**Asymptotic Form.** Dropping constants and lower-order terms, the dominant budget complexity becomes

$$\mathbb{E}[B_{\text{run}}(\rho)] \approx \underbrace{\mathcal{O}(p_v n'_s{}^3)}_{\text{Generation Budget}} + \underbrace{\mathcal{O}(\mathbb{I}[\rho > 0] p_v n'_s{}^2)}_{\text{Pruning Budget}},$$

## E.2 Budget Controlling

For each pruning ratio  $\rho$ , we would like the total compute used by Selective TTS to be comparable to a baseline system with no pruning. Let  $B^*$  denote the target baseline budget at  $\rho = 0$  (either measured in LLM calls or output tokens, depending on the experiment). For a fixed  $\rho$ , let  $b_{\rho,i}$  be the budget consumed by the  $i$ -th run, and let

$$B_\rho^{(n)} = \sum_{i=1}^n b_{\rho,i}.$$

denote the cumulative budget after  $n$  runs. Our goal is to choose a number of runs  $n_\rho^*$  such that  $B_\rho^{(n_\rho^*)}$  is as close as possible to  $B^*$ .

**Step 1: Initial approximation.** For each pruning ratio  $\rho$ , we first obtain a rough estimate of the typical per-run cost  $b_{\rho,i}$  from a small number of pilot runs.<sup>3</sup> Based on this estimate, we select an initial number of runs  $n_0$  (smaller than the baseline run count) and execute these runs, yielding an initial cumulative budget

$$B_\rho^{(n_0)} = \sum_{i=1}^{n_0} b_{\rho,i},$$

which is chosen such that it under-approximates but remains close to the target budget:

$$B_\rho^{(n_0)} < B^* \quad \text{and} \quad B_\rho^{(n_0)} \approx B^*.$$

**Step 2: Incremental refinement.** We initialize

<sup>3</sup>In practice, this can be as simple as running a few preliminary runs and taking their empirical statistics.

the refinement phase with  $n = n_0$  and track the remaining budget gap

$$\Delta^{(n)} = B^* - B_\rho^{(n)}.$$

We then add runs one at a time:

$$B_\rho^{(n+1)} = B_\rho^{(n)} + b_{\rho,n+1},$$

**continuing this process** until we reach the first index  $n^+$  for which  $B_\rho^{(n^+)} \geq B^*$ . Let  $n^- = n^+ - 1$  denote the previous index, for which  $B_\rho^{(n^-)} < B^*$ . **Step 3: Choosing the final run count.** We then select the run count whose cumulative budget is closest to the target:

$$n_\rho^* = \arg \min_{n \in \{n^-, n^+\}} |B_\rho^{(n)} - B^*|.$$

In other words, we compare  $B_\rho^{(n^-)}$  and  $B_\rho^{(n^+)}$  and keep whichever is nearer to  $B^*$ . This procedure ensures that, for each pruning ratio  $\rho$ , the total compute  $B_\rho^{(n_\rho^*)}$  is tightly matched to the baseline budget  $B^*$ , while remaining deterministic and reproducible. We apply the same procedure both when budgeting in terms of LLM calls and when budgeting in terms of output tokens (cf. Fig. 11).

## E.3 Detailed Results for Main Experiments

Table 4 reports the full quantitative results for Selective TTS under different pruning ratios  $\rho$  on the VIS Publication dataset. These values correspond to the performance trends visualized in Fig. 5, and include the number of runs executed, the total number of final reports generated, the average score and standard deviation, and the compute budgets allocated to generation and pruning.

As pruning becomes stronger, the number of runs increases while the number of final reports decreases, reflecting the shift from within-run breadth to cross-run exploration under a matched compute budget. Mean scores improve steadily from  $\rho=0$  to  $\rho=0.6$  (from 61.64 to 65.86), accompanied by a substantial reduction in variance (from 13.36 to 8.16), indicating both higher quality and greater stability. At  $\rho=0.8$ , average scores decline slightly and variance rises, consistent with the risk of over-pruning discussed in the main text.

## E.4 Detailed Results for Ablation Study

Fig. 6 visualizes the comparative behavior of Selective TTS, Random Pruning, and Insight-only Pruning across pruning ratios  $\rho$ . Tables 5, 6, and

Setting	Runs	Total Final Reports	Score (Avg. $\pm$ Std.)	Gen. Budget	Prune Budget
Baseline ( $\rho = 0$ )	8	805	61.20 $\pm$ 13.02	1435	0
$\rho = 0.2$	12	644	63.08 $\pm$ 10.78	1270	218
$\rho = 0.4$	24	498	65.52 $\pm$ 8.29	1211	251
$\rho = 0.6$	45	324	<b>65.70 <math>\pm</math> 8.10</b>	1146	294
$\rho = 0.8$	110	110	65.08 $\pm$ 7.97	1100	330

Table 5: Selective TTS under different pruning ratios  $\rho$  on the VIS Publication dataset (small-scale setting).

Setting	Runs	Total Final Reports	Score (Avg. $\pm$ Std.)	Gen. Budget	Prune Budget
Baseline ( $\rho = 0$ )	8	805	61.20 $\pm$ 13.02	1435	0
$\rho = 0.2$	16	696	62.05 $\pm$ 12.74	1418	0
$\rho = 0.4$	28	573	60.88 $\pm$ 13.09	1443	0
$\rho = 0.6$	60	382	58.59 $\pm$ 14.49	1438	0
$\rho = 0.8$	144	144	59.75 $\pm$ 14.06	1440	0

Table 6: Random pruning baseline under different pruning ratios  $\rho$  (small-scale setting).

Setting	Runs	Total Final Reports	Score (Avg. $\pm$ Std.)	Gen. Budget	Prune Budget
Baseline ( $\rho = 0$ )	8	805	61.20 $\pm$ 13.02	1435	0
$\rho = 0.2$	9	608	60.34 $\pm$ 14.59	1223	152
$\rho = 0.4$	13	564	60.18 $\pm$ 14.47	1323	188
$\rho = 0.6$	12	410	61.73 $\pm$ 14.12	1225	205
$\rho = 0.8$	14	244	62.60 $\pm$ 12.98	1207	244

Table 7: Insight-only pruning under different pruning ratios  $\rho$  (small-scale setting).

Setting	Runs	Total Final Reports	Score (Avg. $\pm$ Std.)	Gen. Tokens	Prune Tokens
Baseline ( $\rho = 0$ )	15	1435	61.64 $\pm$ 13.36	2545971	0
$\rho = 0.2$	24	1232	63.43 $\pm$ 10.48	2357895	215323
$\rho = 0.4$	46	1020	64.97 $\pm$ 9.01	2309491	260884
$\rho = 0.6$	93	688	<b>65.97 <math>\pm</math> 8.19</b>	2209659	321396
$\rho = 0.8$	221	221	65.12 $\pm$ 8.59	2190862	353504

Table 8: Robustness to token-level budgeting: results under matched output-token budgets.

Setting	Runs	Total Final Reports	Score (Avg. $\pm$ Std.)	Gen. Budget	Prune Budget
<b>Q-G Backbone</b>					
Baseline ( $\rho = 0$ )	8	805	61.20 $\pm$ 13.02	1435	0
$\rho = 0.2$	12	644	63.08 $\pm$ 10.78	1270	218
$\rho = 0.4$	24	498	65.52 $\pm$ 8.29	1211	251
$\rho = 0.6$	45	324	<b>65.70 <math>\pm</math> 8.10</b>	1146	294
$\rho = 0.8$	110	110	65.08 $\pm$ 7.97	1100	330
<b>G-Q Backbone</b>					
Baseline ( $\rho = 0$ )	8	788	70.18 $\pm$ 6.87	1404	0
$\rho = 0.2$	12	586	70.83 $\pm$ 6.19	1180	202
$\rho = 0.4$	22	450	71.93 $\pm$ 5.91	1144	234
$\rho = 0.6$	47	302	72.46 $\pm$ 6.64	1124	286
$\rho = 0.8$	108	108	<b>74.60 <math>\pm</math> 5.76</b>	1080	324
<b>Q-Q Backbone</b>					
Baseline ( $\rho = 0$ )	8	650	71.53 $\pm$ 6.08	1172	0
$\rho = 0.2$	10	476	73.02 $\pm$ 6.24	969	165
$\rho = 0.4$	20	375	73.25 $\pm$ 5.51	957	196
$\rho = 0.6$	38	250	74.95 $\pm$ 5.78	930	236
$\rho = 0.8$	90	90	<b>76.06 <math>\pm</math> 5.38</b>	900	270
<b>G-G Backbone</b>					
Baseline ( $\rho = 0$ )	8	747	59.68 $\pm$ 13.67	1344	0
$\rho = 0.2$	11	590	60.76 $\pm$ 13.74	1171	201
$\rho = 0.4$	19	467	63.60 $\pm$ 10.95	1118	233
$\rho = 0.6$	41	304	65.60 $\pm$ 8.99	1072	276
$\rho = 0.8$	103	103	<b>66.87 <math>\pm</math> 8.14</b>	1030	309

Table 9: Cross-model robustness: generation-judge backbone combinations (Q-G, G-Q, Q-Q, G-G).

Setting	Runs	Total Final Reports	Score (Avg. $\pm$ Std.)	Gen. Budget	Prune Budget
<b>Temperature = 0.5</b>					
Baseline ( $\rho = 0$ )	8	675	61.38 $\pm$ 14.10	1224	0
$\rho = 0.2$	10	500	63.29 $\pm$ 12.48	990	170
$\rho = 0.4$	18	438	<b>65.53 <math>\pm</math> 7.95</b>	1024	214
$\rho = 0.6$	37	280	65.08 $\pm$ 9.22	970	250
<b>Temperature = 0.7</b>					
Baseline ( $\rho = 0$ )	8	690	60.71 $\pm$ 13.61	1253	0
$\rho = 0.2$	12	560	64.46 $\pm$ 9.88	1120	192
$\rho = 0.4$	19	417	64.59 $\pm$ 10.10	1008	209
$\rho = 0.6$	39	282	<b>65.39 <math>\pm</math> 8.82</b>	998	256
<b>Temperature = 1.0 (Original)</b>					
Baseline ( $\rho = 0$ )	8	805	61.20 $\pm$ 13.02	1435	0
$\rho = 0.2$	12	644	63.08 $\pm$ 10.78	1270	218
$\rho = 0.4$	24	498	65.52 $\pm$ 8.29	1211	251
$\rho = 0.6$	45	324	<b>65.70 <math>\pm</math> 8.10</b>	1146	294
<b>Temperature = 1.3</b>					
Baseline ( $\rho = 0$ )	6	87	55.14 $\pm$ 18.39	212	0
$\rho = 0.2$	8	54	59.95 $\pm$ 16.33	182	36
$\rho = 0.4$	7	40	62.29 $\pm$ 12.85	167	39
$\rho = 0.6$	11	28	<b>65.34 <math>\pm</math> 8.62</b>	165	45

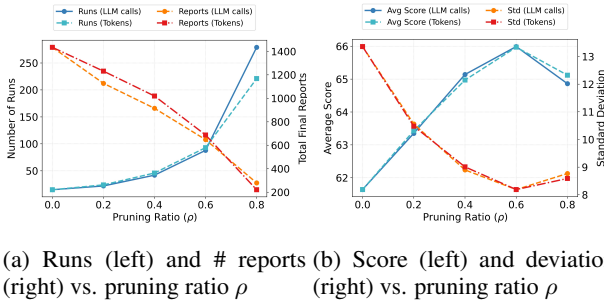
Table 10: Robustness to decoding variation under different temperatures. All results exclude the  $\rho = 0.8$  setting due to instability and high-error execution at extreme temperatures.

Setting	Runs	Total Final Reports	Score (Avg. $\pm$ Std.)	Gen. Budget	Prune Budget
<b>Medical Insurance Dataset (Abdelghany, 2021)</b>					
Baseline ( $\rho = 0$ )	8	760	62.66 $\pm$ 10.72	1293	0
$\rho = 0.2$	10	556	63.98 $\pm$ 10.76	1060	184
$\rho = 0.4$	19	450	64.71 $\pm$ 9.57	1059	221
$\rho = 0.6$	43	292	67.06 $\pm$ 8.07	1028	264
$\rho = 0.8$	99	99	<b>69.23 <math>\pm</math> 7.49</b>	990	297
<b>Diabetes Dataset (Khare, 2020)</b>					
Baseline ( $\rho = 0$ )	8	725	65.74 $\pm$ 10.10	1240	0
$\rho = 0.2$	10	520	66.02 $\pm$ 11.03	1015	175
$\rho = 0.4$	20	417	67.60 $\pm$ 9.19	999	208
$\rho = 0.6$	40	286	68.53 $\pm$ 7.34	999	257
$\rho = 0.8$	95	95	<b>69.78 <math>\pm</math> 6.99</b>	950	285
<b>Education Performance Dataset (Fatima, 2023)</b>					
Baseline ( $\rho = 0$ )	8	800	62.37 $\pm$ 10.97	1352	0
$\rho = 0.2$	11	584	63.12 $\pm$ 10.18	1118	194
$\rho = 0.4$	21	474	65.24 $\pm$ 8.79	1108	232
$\rho = 0.6$	42	310	<b>66.89 <math>\pm</math> 7.59</b>	1070	276
$\rho = 0.8$	104	104	66.69 $\pm$ 8.47	1040	312
<b>Customer Shopping Behavior Dataset (Siddiq, 2022)</b>					
Baseline ( $\rho = 0$ )	8	805	62.37 $\pm$ 11.51	1380	0
$\rho = 0.2$	13	580	63.63 $\pm$ 9.99	1141	197
$\rho = 0.4$	21	474	63.57 $\pm$ 10.75	1136	236
$\rho = 0.6$	45	308	64.30 $\pm$ 10.22	1102	282
$\rho = 0.8$	106	106	<b>65.38 <math>\pm</math> 9.84</b>	1060	318
<b>World Happiness Report Dataset (Buttar, 2023)</b>					
Baseline ( $\rho = 0$ )	8	715	65.15 $\pm$ 8.98	1283	0
$\rho = 0.2$	10	552	65.98 $\pm$ 8.46	1082	186
$\rho = 0.4$	19	429	65.22 $\pm$ 9.37	1038	215
$\rho = 0.6$	41	282	67.21 $\pm$ 8.16	1023	261
$\rho = 0.8$	99	99	<b>70.72 <math>\pm</math> 8.51</b>	990	297

Table 11: Robustness of Selective TTS across five additional datasets under different pruning ratios  $\rho$ .

Setting	Runs	Total Final Reports	Score (Avg. $\pm$ Std.)	Gen. Budget	Prune Budget	Avg. Insight Tokens
<b>Original Prompt</b>						
Baseline ( $\rho = 0$ )	8	805	61.20 $\pm$ 13.02	1435	0	84.50 $\pm$ 19.62
$\rho = 0.2$	12	644	63.08 $\pm$ 10.78	1270	218	85.74 $\pm$ 19.09
$\rho = 0.4$	24	498	65.52 $\pm$ 8.29	1211	251	88.33 $\pm$ 20.15
$\rho = 0.6$	45	324	<b>65.70 <math>\pm</math> 8.10</b>	1146	294	91.78 $\pm$ 21.01
$\rho = 0.8$	110	110	65.08 $\pm$ 7.97	1100	330	94.24 $\pm$ 20.54
<b><math>\sim 130</math> tokens</b>						
Baseline ( $\rho = 0$ )	8	795	65.80 $\pm$ 13.34	1401	0	129.72 $\pm$ 23.08
$\rho = 0.2$	12	608	69.74 $\pm$ 8.54	1216	208	134.56 $\pm$ 23.61
$\rho = 0.4$	20	480	69.58 $\pm$ 8.81	1146	238	135.11 $\pm$ 24.63
$\rho = 0.6$	43	322	71.35 $\pm$ 6.93	1118	288	139.49 $\pm$ 25.58
$\rho = 0.8$	108	108	<b>71.77 <math>\pm</math> 6.73</b>	1080	324	147.11 $\pm$ 26.55
<b><math>\sim 180</math> tokens</b>						
Baseline ( $\rho = 0$ )	8	740	71.15 $\pm$ 10.51	1313	0	177.74 $\pm$ 35.48
$\rho = 0.2$	12	552	72.23 $\pm$ 10.06	1110	190	174.76 $\pm$ 35.72
$\rho = 0.4$	19	465	73.81 $\pm$ 7.24	1093	228	176.82 $\pm$ 35.02
$\rho = 0.6$	41	300	74.50 $\pm$ 6.69	1055	271	191.45 $\pm$ 40.93
$\rho = 0.8$	101	101	<b>74.83 <math>\pm</math> 7.11</b>	1010	303	196.65 $\pm$ 37.53
<b><math>\sim 220</math> tokens</b>						
Baseline ( $\rho = 0$ )	10	790	74.06 $\pm$ 10.44	1416	0	222.87 $\pm$ 51.68
$\rho = 0.2$	15	604	74.97 $\pm$ 8.78	1226	210	220.49 $\pm$ 54.22
$\rho = 0.4$	25	472	76.84 $\pm$ 6.39	1154	240	236.39 $\pm$ 62.93
$\rho = 0.6$	47	314	77.41 $\pm$ 5.59	1126	288	242.31 $\pm$ 58.91
$\rho = 0.8$	109	109	<b>78.96 <math>\pm</math> 5.77</b>	1090	327	245.32 $\pm$ 55.07

Table 12: Robustness to prompt-controlled insight length variation under different insight-generation constraints.



(a) Runs (left) and # reports (right) vs. pruning ratio  $\rho$  (b) Score (left) and deviation (right) vs. pruning ratio  $\rho$

Figure 11: Effects of Selective TTS with LLM-calls as budget and Token budget

7 provide the corresponding numerical results, including the number of executed runs, total final reports, average scores, and compute budgets.

Across all pruning ratios, Selective TTS consistently achieves higher mean scores and lower variance than both ablation baselines, even under the small-scale setting. Random Pruning shows no clear improvement trend and exhibits substantially higher variance, indicating that unguided reduction of candidates is ineffective for quality preservation. Insight-only Pruning performs slightly better than Random Pruning but remains consistently weaker than Selective TTS, demonstrating that pruning applied only at the final stage is insufficient to realize the benefits of structured test-time selection.

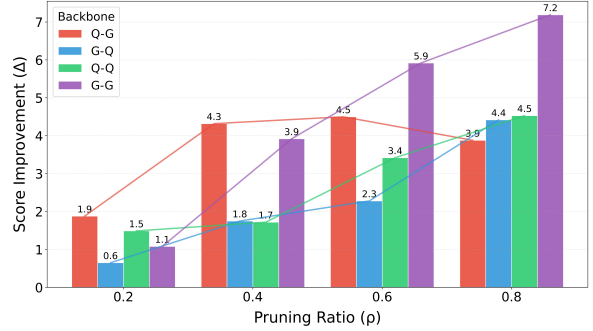


Figure 12: Score improvements ( $\Delta$ ) achieved by Selective TTS across four generator-judger backbone combinations (Q-G, G-Q, Q-Q, and G-G).

## E.5 Detailed Results for Robustness Analyses

This section provides the full numerical results corresponding to the robustness studies reported in §4.4. Across five dimensions, including budget selection and accounting, model backbone, decoding configuration, dataset generalization, and prompt-controlled variation in base insight length, we observe that Selective TTS produces highly consistent scaling trends, supporting the reliability and generality of the method.

**Token-Level vs. LLM-Call Budget.** To ensure that using LLM calls as the compute budget does not distort the observed scaling behavior, we re-run the full set of experiments while controlling the total number of output tokens instead of the

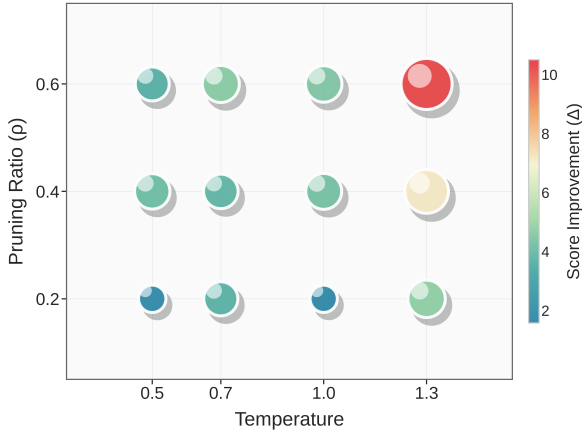


Figure 13: Score improvements of Selective TTS across different decoding temperatures and pruning ratios  $\rho$ . Each bubble represents the improvement over the baseline ( $\rho=0$ ), with color and size indicating the magnitude of the gain.

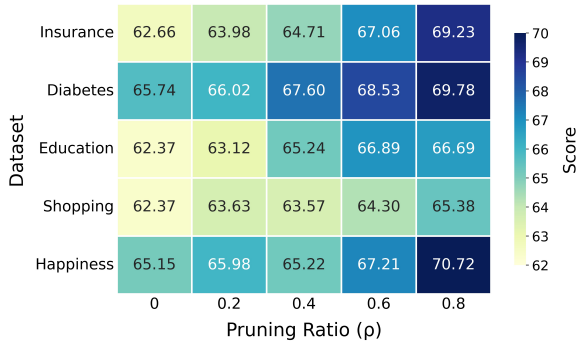


Figure 14: Score heatmap across datasets and pruning ratios  $\rho$ . Selective TTS shows consistent performance gains across all datasets.

number of model invocations. Fig. 11 visualizes the resulting trends in average score and variance when compute is measured at the token level, while Table 8 reports detailed statistics, including the number of runs, the number of final reports, and the corresponding token-level budgets.

Across all pruning ratios  $\rho$ , the qualitative behavior under token-level budgeting closely mirrors that observed under LLM-call budgeting. In particular, mean performance increases from  $\rho = 0$  to  $\rho = 0.6$  before slightly declining at  $\rho = 0.8$ , and the variance follows the same decreasing-then-increasing pattern. The number of runs and the total number of generated tokens also exhibit highly similar trends across pruning ratios, indicating comparable exploration–exploitation trade-offs under both budgeting schemes.

These results confirm that the qualitative scaling behavior of Selective TTS is preserved when com-

pute is measured at the token level. Consequently, using LLM calls as the compute budget provides a faithful proxy for token-level compute. Moreover, LLM-call budgeting is easier to control and reproduce in practice, as token counts can vary substantially with prompt verbosity, chart complexity, and model-specific generation behavior, whereas model invocations offer a more stable and environment-agnostic unit of computation.

**Cross-Model Variation.** To assess the sensitivity of Selective TTS to the choice of model backbone and to examine whether its scaling behavior depends on a particular model family, we evaluate four generator–judger backbone combinations across two model families: Q–Q, Q–G, G–Q, and G–G. Here, Q denotes Qwen2.5-VL-32B-Instruct and G denotes GPT-4.1-nano.

For each generator–judger pairing, we first run the baseline system without pruning ( $\rho = 0$ ) for *eight* runs in order to estimate a matched compute budget. We then apply Selective TTS under higher pruning ratios while keeping the overall compute comparable. Fig. 12 presents the grouped-bar comparison across pruning ratios, showing that all four backbone configurations exhibit consistent performance improvements as  $\rho$  increases. Although the absolute score levels and gain magnitudes vary slightly across model families, the qualitative scaling trends remain highly similar.

Table 9 provides the full numerical breakdown for each backbone pairing and pruning ratio. Taken together, these results indicate that stronger pruning reliably improves quality across all generator–judger combinations. This consistency suggests that the observed gains from Selective TTS generalize across model families and are not driven by any specific generator, judger, or family-specific bias.

**Sensitivity to Decoding Parameters.** Selective TTS benefits from diverse candidates, as pruning is effective only when the search space contains meaningful variation. Accordingly, we use high-diversity decoding by default (temperature = 1.0, top- $p = 0.9$ ) to avoid overly deterministic generation.

To evaluate sensitivity to decoding choices, we vary the generator temperature while keeping top- $p$  fixed, and apply Selective TTS under the same set of pruning ratios. For each temperature setting, we first run the baseline system without pruning

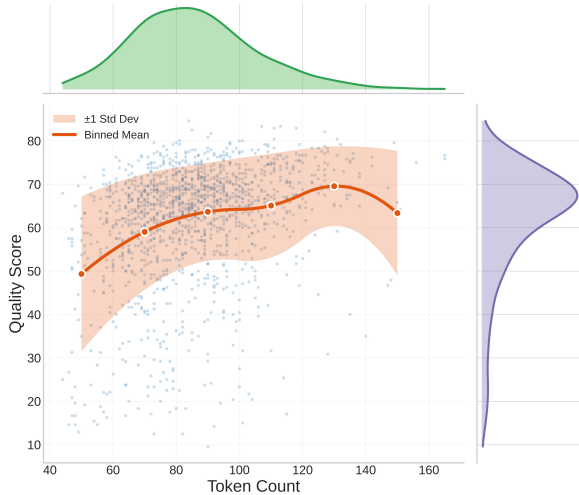


Figure 15: Insight length vs quality score.

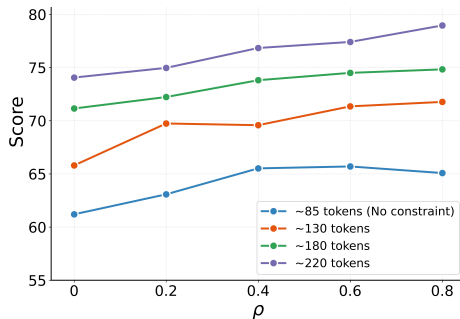


Figure 16: Scaling behavior of Selective TTS under different insight-length regimes.

( $\rho = 0$ ) for *eight* runs to estimate a matched compute budget, and then apply Selective TTS at higher pruning ratios under this fixed budget. Fig. 13 summarizes the resulting score trends across temperatures, while Table 10 reports detailed statistics for each  $\rho \in \{0, 0.2, 0.4, 0.6\}$ .

Across temperatures 0.5, 0.7, 1.0, and 1.3, Selective TTS consistently improves over the baseline, with higher pruning ratios yielding larger gains. Although a modest dip in performance is observed at temperatures = 0.5, likely due to reduced diversity weakening stage-local discrimination, the overall scaling pattern remains stable. Even as generation becomes more deterministic at lower temperatures, Selective TTS continues to provide consistent improvements. At extremely high temperatures, execution errors increase and the number of usable runs decreases, as detailed in Table 10.

Overall, these results indicate that Selective TTS does not rely on a narrow decoding regime and remains robust as the candidate distribution shifts from high- to low-diversity generation.

**Generalization Across Datasets.** To evaluate generalization beyond the VIS Publication dataset, we assess Selective TTS on multiple additional tabular datasets spanning diverse semantic domains and statistical characteristics, including *Medical Insurance* (Abdelghany, 2021), *Diabetes* (Khare, 2020), *Education Performance* (Fatima, 2023), *Customer Shopping Behavior* (Siddiq, 2022), the *World Happiness Report* (Buttar, 2023), and the *Student* dataset. Detailed dataset descriptions are provided in Appendix §B. For each dataset, we first run the baseline system without pruning ( $\rho = 0$ ) for eight runs to estimate a matched compute budget, and then apply Selective TTS under higher pruning ratios.

Fig. 14 visualizes the score trends across pruning ratios for all datasets, and Table 11 reports the full numerical results. Across datasets, Selective TTS consistently improves performance as the pruning ratio  $\rho$  increases from 0 to 0.6, closely mirroring the scaling behavior observed on the VIS dataset. Moderate pruning ( $\rho \in \{0.2, 0.4, 0.6\}$ ) yields steady and reliable gains across all datasets, indicating that the observed scaling behavior is not dataset-specific.

Under the most aggressive setting ( $\rho = 0.8$ ), outcomes become dataset-dependent: while several datasets continue to improve, others show weaker gains. This suggests that excessive pruning may over-eliminate promising branches when the data are noisier or exhibit more complex structure. Overall, these results indicate that Selective TTS generalizes robustly across heterogeneous data domains.

**Sensitivity to Insight Length.** Under the baseline setting ( $\rho = 0$ ), we observe a clear positive correlation between insight length and final evaluation score: longer insights generally receive higher scores, likely because they provide richer context, clearer reasoning, and more supporting detail. Fig. 15 illustrates this association at the instance level, motivating an investigation into whether Selective TTS remains effective as the length of generated insights varies.

To disentangle length effects from scaling behavior, we control the average insight length through prompt-level sentence constraints rather than explicit token limits (e.g., requiring at least six complete sentences with fully developed reasoning), and group the resulting outputs into coarse length regimes. For each regime, we run the baseline system without pruning ( $\rho = 0$ ) for *eight* runs to esti-

mate a matched compute budget and average output length, and then apply Selective TTS under higher pruning ratios. Table 12 reports the detailed numerical results for each length setting, while Fig. 16 summarizes the corresponding performance trends.

Across all insight-length regimes, Selective TTS consistently improves performance as the pruning ratio  $\rho$  increases, with highly similar scaling trends observed across lengths. Although higher pruning ratios tend to select slightly longer insights on average, this effect alone does not explain the observed gains. First, as shown in §4.3, pruning applied only at the final insight stage fails to achieve comparable or stable improvements, indicating that length-based selection at the last stage is insufficient. Second, within each length regime, the variance in average insight length across  $\rho \in \{0, 0.4\}$  remains small (e.g., approximately 4 tokens for the  $\sim 85$ -token setting, 5 tokens for  $\sim 130$  tokens, 1 token for  $\sim 180$  tokens, and 13 tokens for  $\sim 220$  tokens), yet meaningful performance improvements are still observed.

Overall, these results indicate that the gains from Selective TTS cannot be attributed solely to producing longer outputs. Instead, Selective TTS remains effective across a wide range of insight lengths, demonstrating robustness to variation in output verbosity.

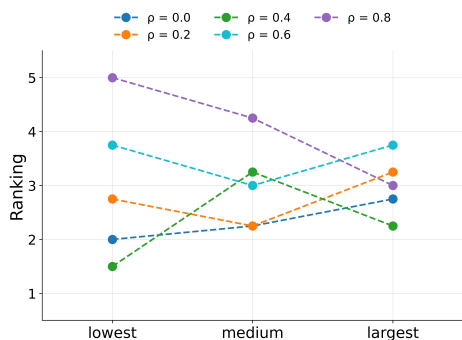


Figure 17: Mean annotator rankings across pruning ratios. Rankings converge as high-scoring report become comparably strong. (higher is better)

## F Supplementary Examples for Qualitative Analysis

To complement the human evaluation in §4.5, we provide additional qualitative examples to illustrate how report quality evolves across different pruning ratios ( $\rho$ ) and score levels. Specifically, we group reports into **low-**, **middle-**, and **high-quality** sets by sampling from the 0%, 50%, and 100% per-

centiles of each  $\rho$ 's score distribution, respectively, and jointly annotate them across pruning configurations. Our analysis primarily focuses on the *low-* and *high-quality* groups, which exhibit clearer contrasts in both model judgment and human perception. These examples further demonstrate how Selective TTS raises the lower bound of report quality while maintaining consistency among top-performing outputs.

### Cross-pruning comparisons (low-quality group).

Fig. 18 compares low-scoring reports across all pruning ratios. The differences are more pronounced than in the high-quality group (Fig. 19). At small  $\rho$ , reports are verbose and generic, whereas increasing  $\rho$  improves logical coherence, factual grounding, and domain awareness. Annotator rankings within the low-quality group consistently favored reports generated at  $\rho=0.8$ , reflecting their relatively higher coherence and reasoning quality. Notably, even the lowest-quality reports at  $\rho=0.8$  outperform their counterparts at  $\rho=0$  or 0.2, indicating a substantial upward shift of the quality floor. At the same time, annotator ranking results show larger dispersion among these low-quality samples (Fig. 17), suggesting that raters could easily distinguish their relative strengths.

### Cross-pruning comparisons (high-quality group).

In contrast, Fig. 19 shows high-scoring reports sampled from each pruning ratio. Their content diversity remains high, but overall quality converges, and each report presents well-grounded reasoning with minor stylistic or focus differences (e.g.,  $\rho=0.8$  emphasizes citation inconsistencies in outlier papers, while  $\rho=0.4$  analyzes paper-length anomalies). Annotators rated these reports comparably strong, as reflected in the small variance of mean rankings in Fig. 17.

## G Qualitative Report Examples on Additional Datasets

To complement the quantitative analyses, we provide qualitative examples of reports generated by our multi-agent pipeline on several additional tabular datasets. These examples illustrate the types of insights and explanations produced by our method across diverse domains beyond the VIS publication dataset.

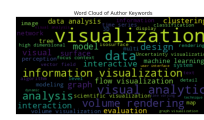
Fig. 20 shows representative reports generated on the Education Performance dataset. Fig. 21 presents example reports from the Medical Insur-



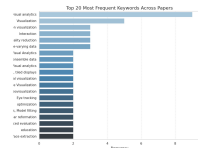
The recurring theme of 'analytics' and 'clustering' points towards an emphasis on statistical and exploratory data analysis. Clustering specifically appears to be a focal area, suggesting that categorizing and grouping data points is a significant task in many studies. This might be due to the complexity of modern datasets requiring advanced segmentation techniques for better understanding. For stakeholders, this implies a need to invest in algorithms and tools that can efficiently handle clustering tasks. An appropriate course of action would be to allocate budget for implementing specialized clustering software, aiming to achieve at least 80% accuracy in cluster identification within new projects by year-end.



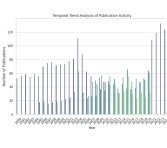
The inclusion of 'machine learning', 'graph', and 'network' signals a growing convergence between visualization and AI-driven approaches. This crossover likely reflects efforts to leverage machine learning for automatic data analysis and visualization generation, indicating a future where visualizations become increasingly automated and intelligent. Organizations specializing in data science could benefit from exploring partnerships with machine learning experts to develop hybrid solutions. **\*\*Implication\*\***: Data visualization tool vendors should plan to integrate machine learning APIs into their platforms, aiming to automate 80% of routine data visualization tasks by the end of the current fiscal year, thereby reducing manual effort and increasing efficiency.



The inclusion of terms such as 'Uncertainty visualization' and 'time varying' indicates a nuanced interest in handling complex data characteristics, such as uncertainty and temporal dynamics. These terms suggest that authors are addressing challenges in visualizing inherently uncertain or dynamic data sets, which could be critical in fields like predictive analytics or real-time monitoring systems. This insight suggests that there is a growing demand for visualization techniques that can effectively communicate variability and change over time. Practitioners could develop new visualization frameworks that explicitly highlight uncertainty ranges and temporal trends, thereby aiding stakeholders in making informed decisions under dynamic conditions.

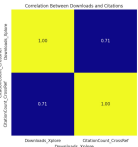


Several keywords, such as 'Dimensionality reduction', 'Time-varying data', and 'Ensemble data', have similar frequencies (~3), indicating that these topics are equally important but distinct areas of inquiry. This suggests a diverse set of challenges and methods being explored in handling large and dynamic datasets. Researchers looking to innovate might explore novel combinations of dimensionality reduction techniques with ensemble data analysis, particularly when dealing with temporal data. Conference organizers or professional associations in the field could plan panel discussions or workshops focusing on these interconnected areas to facilitate knowledge sharing and collaboration among experts.

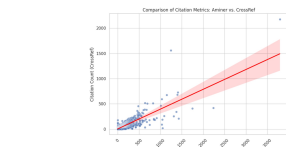


The 'Vis' conference consistently shows the highest number of publications compared to other conferences throughout the entire time period depicted in the chart. This dominance is evident as the blue bar (representing 'Vis') surpasses other conferences like 'InfoVis', 'VAST', and 'SciVis' by a substantial margin, often reaching over 80 publications annually, especially around 2004 and post-2010. This trend suggests that 'Vis' may be more established or have a broader appeal within the research community. To capitalize on this trend, actors responsible for conference organization could consider leveraging the popularity of 'Vis' to boost cross-conference collaborations or organize joint sessions with other venues to enhance their reach and impact.

Figure 18: Low-scoring reports across all pruning ratios. The corresponding scores are:  $\rho = 0$ : 11.3,  $\rho = 0.2$ : 20,  $\rho = 0.4$ : 11.3,  $\rho = 0.6$ : 32.9,  $\rho = 0.8$ : 47.5.



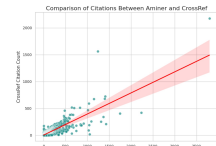
The chart reveals a moderate positive correlation between 'Downloads\_Xplore' and 'CitationCount\_CrossRef', suggesting that higher download counts on the Xplore platform are associated with a higher number of citations in CrossRef databases. The correlation coefficient of 0.71 indicates a significant but not perfect relationship, implying that while downloads may influence citation counts, other factors likely play a role as well. This suggests that platforms like Xplore could be leveraging their user engagement data ('Downloads') to predict or enhance the scholarly impact of published content. Researchers or editors might consider prioritizing promotional strategies for articles with higher initial download rates on Xplore, as these are more likely to gain recognition and citations over time. To further explore this dynamic, future analyses should investigate whether there is a lag period between downloads and subsequent citations.



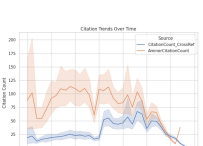
The presence of outlier points, especially those far above or below the regression line, highlights instances where one database significantly undercounts or overcounts citations relative to the other. These outliers may represent edge cases such as interdisciplinary studies, open-access articles, or publications heavily cited outside traditional academic settings. For example, a paper heavily cited in social media or popular science platforms might show inflated CrossRef counts compared to Aminer, which relies more on formal scholarly sources. Identifying such outliers is crucial for understanding their context, as they may reflect unique circumstances or methodological biases in citation tracking. Researchers should scrutinize these instances closely, perhaps by investigating supplementary data or considering alternative metrics, to avoid misinterpretation of citation impacts.



A notable insight from the chart is the significant outlier in paper length observed in the year 2000, where the average page length dramatically peaked at approximately 175 pages, far exceeding the typical range of around 10-15 pages observed before and after this period. This spike suggests a unique event or change in publication practices that year. Considering the otherwise stable trend, this outlier could indicate a specific conference, journal, or policy shift that temporarily encouraged longer submissions. Given the prominence of this anomaly, further investigation into publications from 2000 would be valuable to understand the underlying cause and determine whether similar trends might emerge again. An actionable next step would be for academic editors or organizers to analyze papers from 2000 to identify common themes or factors contributing to this increase.



The scatter plot reveals a positive correlation between Aminer Citation Count and CrossRef Citation Count, as indicated by the upward-sloping red regression line. This suggests that as the citation count in Aminer increases, there is a corresponding increase in the citation count reported by CrossRef. However, the scatter of data points around the regression line implies variability in this relationship, meaning some papers may have citation counts that deviate significantly from the expected trend. For example, while most points are near the line, there are outliers such as the paper with approximately 3700 Aminer citations but only ~400 CrossRef citations. This could indicate discrepancies in how these platforms count citations or differences in their coverage of certain publication types or fields. Given this variability, actors like researchers or publishers might need to consider both citation databases and cross-reference results to ensure a more accurate understanding of a paper's impact.



The Citations from CrossRef and Aminer show a significant disparity over time, with Aminer consistently reporting higher citation counts across most years compared to CrossRef. This divergence becomes more pronounced after 2010, where Aminer's citation count stabilizes around 80-100 citations, while CrossRef remains significantly lower (~20-40 citations). This suggests that Aminer may employ a broader or different methodology in counting citations, possibly capturing sources not indexed by CrossRef. Researchers evaluating study impact should consider using multiple sources for citation counts to ensure a comprehensive understanding of a work's reach. Actionably, publishers and researchers should validate their citation metrics against both sources to mitigate potential biases or incomplete datasets.

Figure 19: High-scoring reports across all pruning ratios. The corresponding scores are:  $\rho = 0$ : 85.4,  $\rho = 0.2$ : 85,  $\rho = 0.4$ : 84.6,  $\rho = 0.6$ : 84.2,  $\rho = 0.8$ : 84.2.

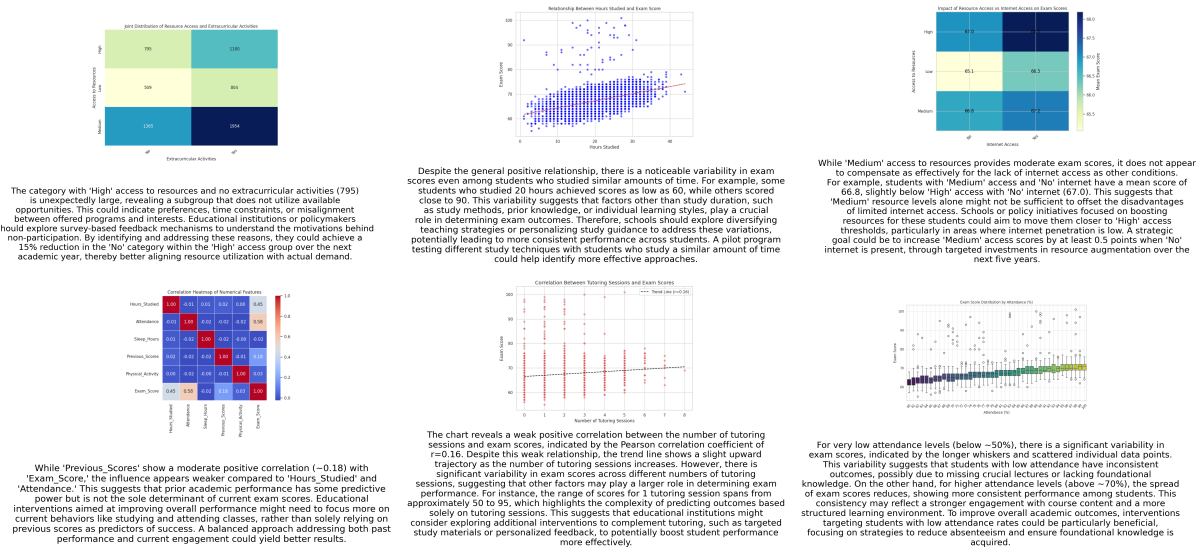


Figure 20: Example reports for Education Performance Dataset

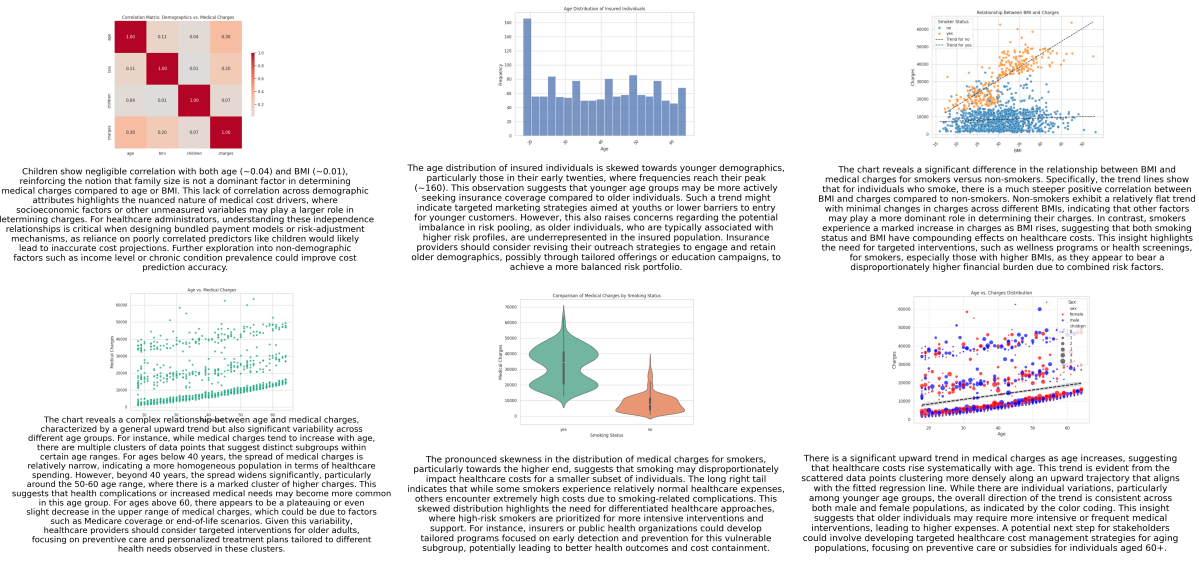
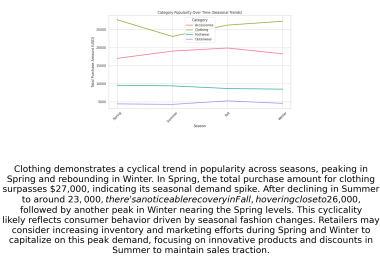
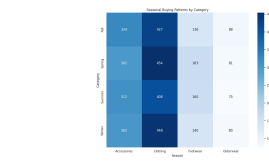


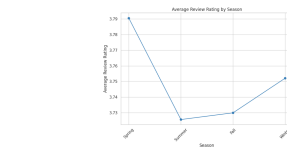
Figure 21: Example reports for Medical Insurance Dataset



Clothing demonstrates a cyclical trend in popularity across seasons, peaking in Spring and rebounding in Winter. In Spring, the total purchase amount for clothing surpasses \$27,000, indicating its seasonal demand spike. After declining in Summer to around \$21,000, there is a noticeable recovery in Fall, hovering just above \$26,000, followed by another peak in Winter nearing the Spring levels. This cyclicity likely reflects consumer behavior driven by seasonal fashion changes. Retailers may consider increasing inventory and marketing efforts during Spring and Winter to capitalize on this peak demand, focusing on innovative products and discounts in Summer to maintain sales traction.



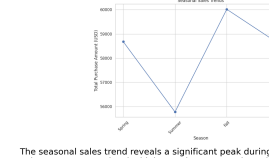
Footwear demand exhibits noticeable seasonal variation, showing higher purchases in Summer (160) and Spring (136), compared to Fall (136) and Winter (140). This suggests that warmer seasons may trigger increased demand for lighter footwear types. Retailers might consider adjusting inventory levels or launching promotions during the warmer months to maximize sales. Conversely, the relatively lower demand in colder seasons may signal the need for different marketing strategies focused on winter boots or heavier footwear options. This seasonal disparity highlights the importance of adapting inventory management and advertising based on weather trends.



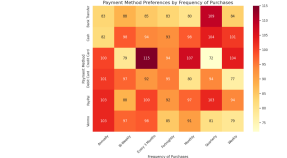
Summer has the lowest average review rating at approximately 3.71, suggesting it might be the most challenging season for maintaining customer satisfaction. This dip could stem from factors like intense competition, user fatigue, or a shift in consumer priorities during the warmer months. Given that summer marks a notable low compared to other seasons, the company might benefit from conducting seasonal analyses to pinpoint the exact causes. For instance, they could survey customers specifically in summer to understand their concerns better. Once identified, actions like implementing summer-exclusive features, improving logistics, or offering seasonal discounts could help lift ratings closer to other seasons' benchmarks.



The payment method landscape indicates a competitive environment among the top methods—Credit Card, PayPal, Cash, and Debit Card—all contributing approximately 17%-18% of total sales. This close parity suggests that customers have diverse preferences, making it critical for businesses to maintain flexibility in their payment options. Given the similarity in usage across these methods, marketing strategies should focus on enhancing user experience and incentives across multiple platforms to capture a larger share of customer preference. The small differences (~1%) between these methods imply that even slight improvements in user convenience or loyalty programs could shift customer choices noticeably.

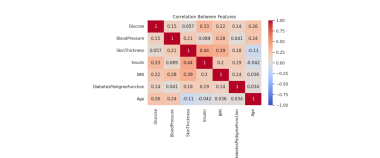


The seasonal sales trend reveals a significant peak during Fall, where the total purchase amount reaches its highest point at approximately \$60,000. This peak is roughly \$4,000 higher than the next highest season (Winter). The sharp drop observed between Spring and Summer, followed by a substantial rise from Summer to Fall, suggests that consumer spending behavior may be heavily influenced by seasonal factors such as holiday shopping or changes in purchasing habits. The similarity in values between Winter (\$8,000) and Spring (\$9,000) indicates a relatively stable period before the summer dip. Retailers could leverage this insight by intensifying marketing campaigns and inventory preparation ahead of Fall, potentially increasing revenue by capitalizing on the peak season. For instance, retailers might consider increasing stock levels or running promotional campaigns in early Fall to capture this demand surge.

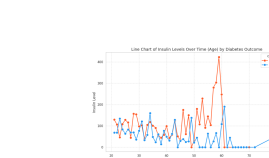


Bank transfers exhibit a periodic pattern, with usage peaking during the bi-weekly and quarterly purchase intervals. These intervals show higher scores (88 and 109, respectively) compared to monthly and weekly intervals. This cyclical behavior may reflect structured payment cycles where consumers prefer bank transfers for routine financial planning. Businesses could leverage this by offering bank transfer discounts during these intervals, potentially increasing transaction volumes by appealing to customers with structured purchasing habits. For instance, promoting bank transfers during bi-weekly and quarterly cycles could see a 10-15% increase in usage within six months.

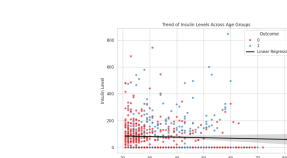
Figure 22: Example reports from Shopping Behavior Dataset



The DiabetesPedigreeFunction exhibits a notably low correlation with most features, including Glucose (0.14), BloodPressure (0.041), SkinThickness (0.18), Insulin (0.19), and BMI (0.14). This finding implies that family history or genetic susceptibility to diabetes, as represented by the DiabetesPedigreeFunction, does not strongly predict other physiological characteristics commonly associated with diabetes onset. This may suggest that environmental or lifestyle factors play a more dominant role than genetics in modulating these traits. Researchers may need to focus on identifying interactions between DiabetesPedigreeFunction and other unmeasured variables to explain its limited standalone predictive power. Public health initiatives could prioritize lifestyle interventions, such as diet and exercise, even in individuals with a positive family history, due to the weak genetic associations observed here.



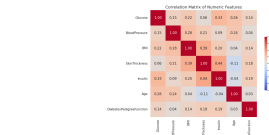
The sharp increase in insulin levels observed in Outcome 1 around age 60 suggests a significant metabolic shift that could indicate the onset of severe diabetic complications. This peak, exceeding 400 units, stands out as an anomaly compared to the consistent trend in Outcome 0, where insulin levels remain relatively stable and decline gradually after peaking earlier. The dramatic spike likely reflects heightened insulin resistance or hyperglycemia, conditions commonly associated with advanced diabetes progression. This finding suggests that individuals with Outcome 1 may benefit from more intensive monitoring and intervention during this critical period, potentially delaying the progression of diabetes-related complications.



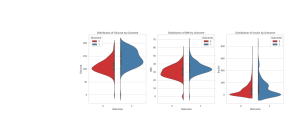
The chart illustrates a substantial overlap in insulin levels between 'Outcome 0' and 'Outcome 1' across various age groups. While there is a discernible separation in younger age groups, the lines for both outcomes become increasingly indistinguishable as age progresses. This overlap becomes particularly pronounced in the 50+ age range, where data points from both groups cluster around very low insulin levels. The lack of clear differentiation between outcomes later in life may suggest that age outweighs other distinguishing factors, potentially indicating shared physiological processes or limitations in distinguishing Outcome 1 from 'Outcome 0' based solely on insulin levels. Policymakers may need to reassess how diagnostic criteria adapt for different age groups to ensure accuracy in identifying potential health risks.



The scatter plot reveals a positive correlation between BMI and Glucose levels for individuals across both outcomes (Outcome 0 and Outcome 1). As BMI increases, there is a general upward trend in glucose levels, suggesting that higher body mass index is likely associated with higher blood sugar levels. This trend is evident through the distribution of data points, which cluster more densely along the upper-right quadrant of the chart where both BMI and Glucose are high. Furthermore, the regression line suggests a consistent increase in glucose levels with increasing BMI, although the relationship appears non-linear, as indicated by the curvature of the data points around the line. This insight implies that interventions targeting weight management may play a crucial role in managing glucose levels, particularly for populations at risk of developing glucose-related health issues. Stakeholders such as healthcare providers could focus on promoting weight loss programs or lifestyle changes for patients with higher BMIs to potentially reduce glucose levels.



Glucose shows moderate correlations with several features: BMI (0.22), BloodPressure (0.15), and Insulin (0.33). This pattern indicates that glucose levels are influenced by a combination of body composition and metabolic activity. The presence of these correlations may highlight the complex interplay between lifestyle factors and physiological responses. For instance, interventions aimed at reducing BMI and managing BloodPressure might indirectly help stabilize glucose levels in pre-diabetic populations. Further studies could investigate whether a multi-faceted approach addressing all correlated factors leads to more effective glucose control compared to single-focus strategies. The threshold for action could be set at identifying individuals with simultaneous elevations in these correlated metrics, targeting early intervention protocols.



The relationship between glucose and insulin levels across the outcomes reveals an indirect linkage tied to metabolic health. While glucose levels differ sharply between outcomes 0 and 1, insulin levels also exhibit a distinct pattern indicative of different physiological states. For outcome 0, the low insulin levels coupled with relatively normal glucose suggest a lack of metabolic stress or insulin resistance. Conversely, the high insulin levels in outcome 1, even though glucose levels are elevated, imply compensatory insulin secretion possibly to counteract insulin resistance. This observation underscores the complex interplay between glucose metabolism and insulin response. Healthcare providers could implement strategies to assess insulin sensitivity concurrently with glucose monitoring, focusing on interventions such as lifestyle modifications or medications to address insulin resistance for patients with outcomes consistent with outcome 1.

Figure 23: Example reports from Diabetes Dataset

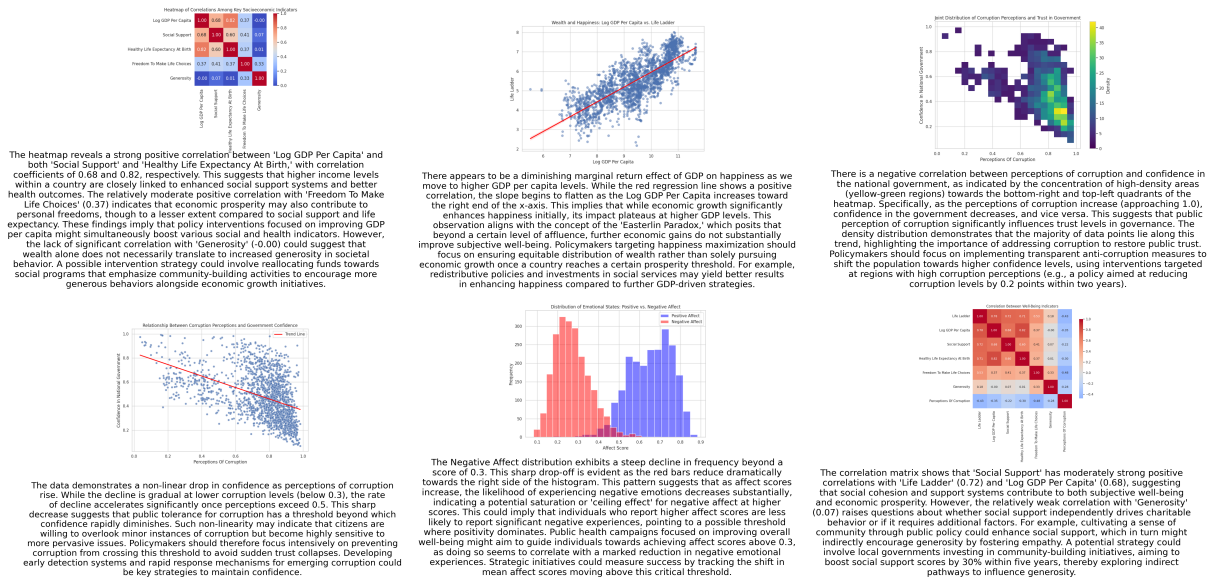


Figure 24: Example reports from World Happiness Report Dataset

ance dataset. Fig. 22 displays reports generated on the Shopping Behavior dataset. Fig. 23 provides examples from the Diabetes dataset. Finally, Fig. 24 shows reports generated on the World Happiness Report dataset.

## H Pipeline Architecture and Stage Outputs

This section provides an extended description of our multi-agent pipeline architecture, its core design components, and the intermediate outputs produced at each stage. We first present an end-to-end overview of the pipeline, detailing how data flows through generation, evaluation, and pruning stages (§H.1). We then illustrate representative example outputs produced by each pipeline stage to concretely demonstrate how intermediate results evolve throughout the pipeline (§H.2). Finally, we describe the prompt design and specifications for all major agents and judges involved in each stage, including both task-level generation and stage-local evaluation modules (§H.3).

### H.1 End-to-End Pipeline Overview

Fig. 25 presents the full workflow of our proposed multi-agent data science pipeline, expanding upon the conceptual stages illustrated earlier in Fig. 2. While Fig. 2 provides a high-level overview of the agents and their interconnections, Fig. 25 offers a more detailed depiction of the end-to-end process, showing how data samples are progressively refined through generation, evaluation, and pruning

across multiple stages.

### H.2 Representative Outputs at Each Pipeline Stage

Here we present an example metadata report (data profile) generated from the VIS Publication dataset.

#### An example of generated metadata report

**Dataset Overview.** The dataset contains 3,877 rows and 20 columns, comprising a mixture of categorical, numerical, and text-based fields. The presence of identifiers, publication metadata, citation statistics, and textual abstracts suggests that this dataset represents a structured collection of academic publications.

**High-Level Description.** Each record corresponds to an academic paper, capturing conference information, publication year, title, DOI and URL links, page ranges, paper type, abstract, author names and affiliations, citation counts, download statistics, awards, and graphical replicability indicators. Overall, the dataset supports analysis of scholarly output, citation impact, author contributions, and publication trends within the visualization research community.

**Schema Summary.** The dataset schema covers a wide range of publication metadata fields. Venue and temporal information are represented by attributes such as Conference (e.g., "VIS") and Year (e.g., 2024). Core content fields include Title (e.g., "Impact of Vertical Scaling on Normal Probability Density Function Plots") and Abstract, which provide textual descriptions of each paper.

Authorship information is captured through AuthorNames and AuthorNames-Deduped (e.g., "Racquel Fygenon; Lace Padilla"), along with AuthorAffiliation (e.g., "Northeastern University, USA"). Document identifiers and access links are recorded via DOI (e.g., "10.1109/TVCG.2024.3456396") and Link (e.g., an IEEE Xplore URL), while page ranges are specified by FirstPage and LastPage (e.g., 984–994).

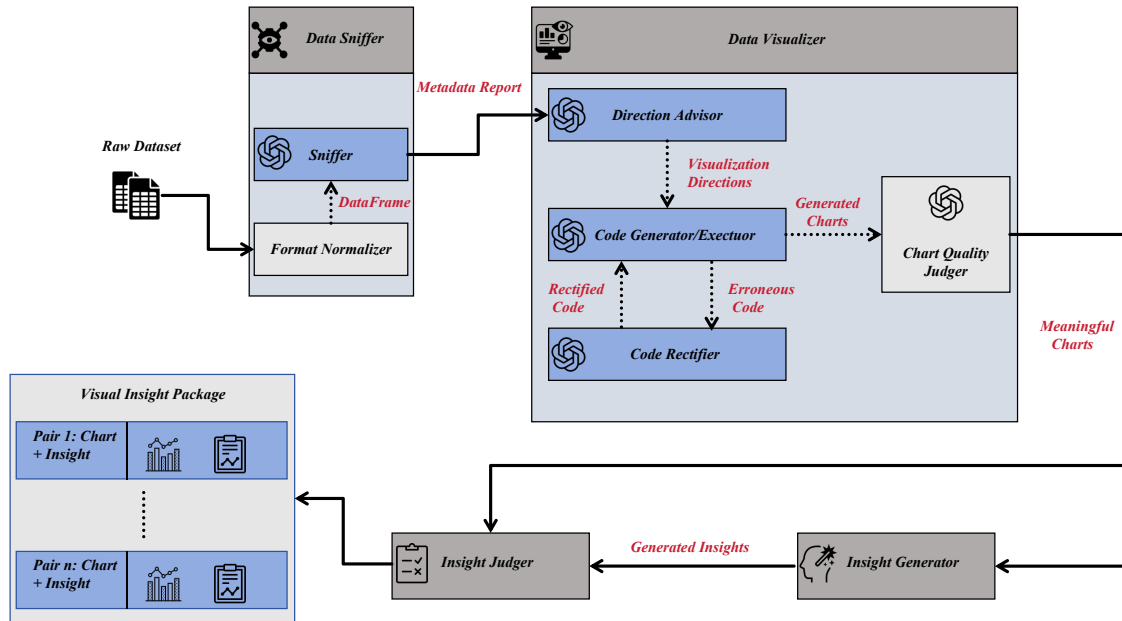


Figure 25: Detailed Pipeline of Insight Generation.

Citation and impact-related attributes include CitationCount\_CrossRef (e.g., 0), PubsCited\_CrossRef (e.g., 60), AminerCitationCount, and Downloads\_Xplore (e.g., 123), enabling analysis of scholarly influence and readership. Additional descriptive fields such as AuthorKeywords (e.g., “visualization, uncertainty, vertical scaling”), PaperType (e.g., “J” for journal article), Award, and GraphicsReplicabilityStamp provide context on paper themes, publication format, recognition, and reproducibility indicators.

**Potential Analysis Directions.** The dataset enables several exploratory analyses, including: (i) publication trends over time and across conferences, (ii) comparisons of paper length and type, and (iii) relationships between citation counts, referenced publications, and impact metrics.

```

    "x_axis": "Year",
    "y_axis": "Number of Papers",
    "color_scheme": "category",
    "sort_by": "descending"
  }
},
...
]

```

Following the visualization direction, the pipeline generates a concrete chart instance, as shown below.

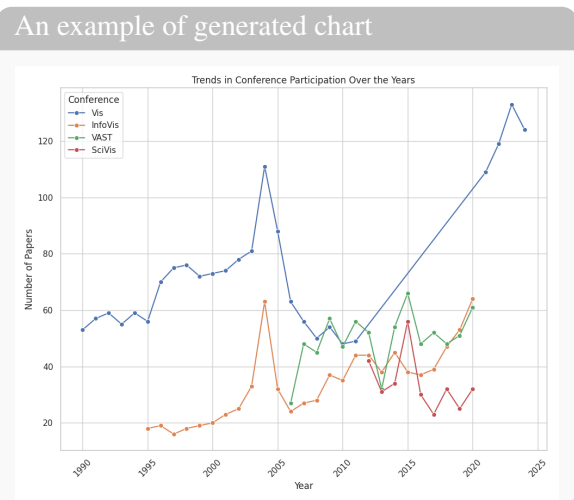
Based on this metadata report, we next show an example visualization direction generated by the visualizer.

An example of generated visualization direction

```

[
  {
    "topic": "Temporal Trend of Publications",
    "chart_type": "line",
    "variables": ["Year", "Conference"],
    "explanation": "This line chart illustrates how the number of publications changes over time across different conferences. By aggregating papers by year and conference, it highlights periods of increased research activity as well as potential declines in specific venues.",
    "parameters": {
      "group_by": "Conference",
      "aggregate_method": "count",

```



Given the generated chart, the insight generator produces a structured textual interpretation grounded in the visual evidence, shown below.

### An example of generated insight

```
{
  "description": "All four conferences demonstrate varying patterns of growth with distinct phases—ranging from early stagnation to significant spikes or gradual increases. Notably, Vis and SciVis both exhibit late-stage growth spurts after 2015, potentially driven by advances in technology and interdisciplinary research. In contrast, InfoVis and VAST show earlier growth but later stabilization or fluctuation, suggesting saturation in certain topics or subdomains. This heterogeneity highlights the diverse maturity levels of different visualization fields. Policy-makers or academic leaders could leverage this insight to strategically allocate resources, supporting newer domains with growth potential while revitalizing mature ones via innovative thematic directions. Targeting a unified growth trajectory across all conferences by 2030 could help maintain balance and synergy within the broader visualization community.",
  "evidence": "Comparing all lines, Vis and SciVis show robust growth post-2015, whereas InfoVis and VAST display stabilization or variability after their respective peaks, highlighting divergent evolutionary paths among the conferences."
}
```

### H.3 Prompt Design and Specification

We present the prompt design for three major components of our pipeline: (1) generation agents for data profiling, visualization, and insight generation, (2) multi-level judges for final evaluation, and (3) stage-local evaluators used in Selective TTS for process-based pruning.

**Data Profiling, Visualization, and Insight Generation Agents.** As illustrated in Fig. 25, the core pipeline comprises six generation components: the *Data Profiling*, *Visualization Direction Generation*, *Code Generation*, *Code Rectification*, *Chart Quality Filtering*, and *Insight Generator*. Each component operates under a modular system prompt that specifies its role, expected input–output format. The complete system prompts for these generation components are presented below.

#### System Prompt: Data Profiling

You are a professional data curator writing an official-style “About Dataset” page (as on Kaggle/Hugging Face). You will ONLY receive the metadata information and sample rows of the dataset: (1) dataset shape (rows x columns); (2) column names with detected data types; (3) 1–2 sample rows. Your task is to generate a clear, structured, and comprehensive dataset introduction strictly based on the provided information. Base ALL content strictly on this input. Do NOT infer anything about missing values, full distributions, correlations, provenance, licensing, collection period, or coverage unless explicitly evident from names or the sample.

##### 1. Your goals

- 1) **Briefly introduce the dataset:** describe what it appears to contain and how it is structured (refer to shape and the mix of data types).
- 2) **Explain each variable in detail:** infer likely meanings

from column names and the sample (use cautious language such as “likely” or “appears to”); mention units or ranges only if clearly suggested (e.g., `_usd`, `_pct`, `lat/lon`, `date`).

3) **Propose potential analysis directions and visualization themes:** suggest exploratory analyses grounded in the schema (e.g., numeric distributions, categorical comparisons, temporal trends, geographic patterns if names imply geo fields, keyword summaries for text).

4) Provide clear, structured documentation that downstream chart planners and code generators can follow immediately.

##### 2. Output format (Markdown)

Use the following sections and formatting in markdown:

###### ## About Dataset

– **Shape:** `<rows> x <cols>`. Briefly describe the schema and mix of types (numeric/categorical/text/datetime).

– **High-level summary:** A concise statement of what the table likely represents (strictly based on names and sample).

###### ## Schema Summary

Output as CSV-style plain text lines using the header:

Column, Detected Type, Example, Likely Meaning, Suggested Role

Each subsequent line corresponds to one column. For example:

`id, integer, "12345", likely an identifier for each record, id`

Where Suggested Role  $\in$  {feature, target, id, timestamp, text, geo, meta}.

###### ## Potential Uses & Analysis Directions

Be concrete and grounded in the provided schema.

– Suggest meaningful exploratory analyses and visualizations.

– If a column appears to be a target variable, note potential modeling tasks.

##### 3. Output rules

– Use clear Markdown headings; for tables, use CSV-like plain text, not Markdown tables.

– Be specific yet cautious: never invent facts beyond names/sample.

– Tone must be professional and documentation-like, as if authored by the dataset creator.

Based on this metadata report, we next show an example visualization direction generated by the visualizer.

#### System Prompt: Visualization Direction

You are a professional data visualization expert. Your task is to propose well-defined visualization directions for a given dataset.

##### You will be provided with:

- Metadata information (number of rows/columns, data types, etc.)
- Sample data from the dataset

##### Requirement:

1. Generate `{num_directions}` **concise, diverse, and actionable** directions for visualizing the dataset.

2. The directions should not overlap in their focus; each direction should focus on a different aspect of the data.

3. Each direction must:

– Focus on a specific aspect of the data (e.g., distribution, correlation, time trend, category comparison, anomaly detection).

– Include **all** of the following keys: `topic`, `chart_type`, `variables`, `explanation`, `parameters`.

– Use only existing column names from the dataset in variables.

– `chart_type` should be a standard, executable visualization type (e.g., “bar”, “line”, “scatter”, “histogram”, “boxplot”, “heatmap”, “wordcloud”).

– `parameters` may include chart-specific options such as sorting, grouping, aggregation method, bins, color scheme, etc.

4. Avoid vague or generic suggestions. Each explanation

should state **why** the chart is relevant and what insights it may reveal.

5. Try to use **different chart types** and cover various analytical angles across the directions. Be creative!

**Output format (all keys mandatory, valid JSON only):**

```
[
  {
    "topic": "...",
    "chart_type": "...",
    "variables": ["...", "..."],
    "explanation": "...",
    "parameters": {"param1": "...", "param2": "..."}
  },
  {
    "topic": "...",
    "chart_type": "...",
    "variables": ["...", "..."],
    "explanation": "...",
    "parameters": {"param1": "...", "param2": "..."}
  }
]
```

**Attention:**

- Output **only** the JSON array inside a pure Markdown code block (“`json ...`”), without any extra text, commentary, or formatting.

- Ensure the JSON is valid and directly parsable.

- Strict adherence to the output format is required.

## System Prompt: Code Generation

You are a Python data visualization code generator. Your task is to generate **production-ready, executable** Python code based on the following inputs: - Metadata information: data types, shape of the dataset, introduction of the dataset and sampled data. - A given visualization direction (topic, chart\_type, variables, parameters).

**Requirements:** 1. The code must: - Use ONLY: import pandas as pd, import matplotlib.pyplot as plt, import seaborn as sns, import numpy as np, import networkx as nx, from wordcloud import WordCloud. - Always set matplotlib backend to "Agg" before importing pyplot. Do not use plt.show(), only save figures to the given path.

- Load the dataset (CSV format) using df = pd.read\_csv({data\_path}).

- Implement the visualization as specified in the given direction.

- Add a descriptive title, x-axis label, and y-axis label. - Apply a consistent style, e.g., sns.set\_theme(style="whitegrid") and plt.figure(figsize=(8, 6)).

- Ensure category labels on the x-axis are rotated if necessary (plt.xticks(rotation=45)).

- Call plt.tight\_layout() before saving.

- Save the plot to plt.savefig({output\_path}) and then call plt.close().

2. The code must be directly executable without modification. 3. Pay attention to data types (categorical vs. numerical). 4. Do **not** create or print any other output besides the plot file.

5. Always import all necessary libraries explicitly (e.g., import seaborn as sns, import pandas as pd, import matplotlib.pyplot as plt).

6. Ensure the code is executable and contains no undefined variables or functions.

**Output format:**

```
```python
# no comments
...(code here)...
```

**Attention:** 1. Output only valid Python code inside a Markdown code block.

2. Do not use any libraries beyond pandas, matplotlib, seaborn, numpy, networkx, and wordcloud.

3. Return only valid, ready-to-run Python code.

4. Ensure syntax correctness and reproducibility.

## System Prompt: Code Rectification

You are a Python code rectifier. Your task is to fix the provided code strictly according to the error feedback so that it runs successfully.

**You will be given:** - The original code that failed - The exact error feedback from execution

**Hard Constraints (NEVER violate):** - Do NOT change any I/O

paths, file names, or file formats that appear in the original code (e.g., dataset path, output image path). - Do NOT introduce new third-party dependencies beyond: pandas, matplotlib.pyplot, seaborn. - Do NOT change the intended chart type or analytical intent unless the error explicitly requires it (e.g., unsupported parameter). - Do NOT print or display extra output (no plt.show, no print); saving the figure is the only side effect. - Keep the code as a single, directly executable script (no notebooks magics, no functions required).

**Fixing Guidelines:** - Resolve import, name, attribute, parameter, dtype, and seaborn/matplotlib version-compat errors (e.g., deprecated args) by using supported alternatives. - If the error is due to a missing column or invalid variable, add a clear ValueError with a helpful message rather than guessing column names. - Ensure the figure is saved exactly to the same output path present in the original code. - Add minimal robustness only when needed to fix the error (e.g., plt.tight\_layout()), and close figures with plt.close() after saving. - Ensure valid Python syntax (no comments, no extra text), and that the script is immediately runnable.

**Allowed libraries:**

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import networkx as nx
from wordcloud import WordCloud
```

**Output format:**

```
```python
# no comments
...(corrected executable code here)...
```

**Attention:** 1. Output only the code inside a pure Markdown code block (python ... ), without any extra text, commentary, or formatting. 2. Ensure the code is syntactically correct and ready to run.

## System Prompt: Chart Quality Filtering

You are an automated reviewer of data visualizations. Given a chart image, determine whether the image is legible and meaningful for analysis. Your judgment must be based primarily on what is visible in the image.

**You will be given:** - A chart image (as a base64-encoded PNG)

**What to evaluate:** 1. **Readability & Clarity:** Axes/titles/legends present and readable; tick labels not overlapping; reasonable tick density; text not clipped; adequate contrast; figure size/aspect sensible; tight\_layout-quality. 2. **Meaningfulness:** Non-empty, non-constant data; visible variation; ordering/sorting makes sense; annotations/units (if relevant) clear; not a trivial or degenerate view (e.g., 100% single category).

**Output format (valid JSON in a single fenced code block; no extra text):**

```
```json
{
  "is_legible": true|false,
  "evidences": [
    "axes and title are readable",
    "labels are not overlapping"
  ]
}
...
```
```

**ATTENTION:** 1. Output only the JSON object inside a pure Markdown code block (json ... ), without any extra text, commentary, or formatting. 2. Use boolean true/false for the is\_legible field. 3. Follow the format strictly and do not add any additional information.

## System Prompt: Insight Generation

You are an expert data analyst with strong visual interpretation skills. I will provide you with a single chart; please generate {num\_insights} **high-quality** insights grounded strictly in what is visible in the chart.

**Requirements:**

- **Observation & Evidence Completeness:** Cover all relevant aspects of the observation, including:

- Subspace: the specific segment/condition/time window

(e.g., “post-2024-05”, “top decile”, “for X>Y”).

- Breakdown: the variables or metrics involved (e.g., “blue line vs orange bars after 2022-Q3”).
- Effect size: estimated size (e.g., “~15-20%”), direction (e.g., “increase”), and type (e.g., “relative”).

- **Traceability:** Every claim must point to exact series, marks, axes, or ranges so a reader can verify it on the chart.
- **Insightfulness:** Use chart cues to expose structure (subgroup heterogeneity, changepoints, seasonality, contribution patterns) and reason about possible causes behind the observation.
- **Non-triviality & Novelty:** Go beyond simple narration; avoid tautologies or trivial descriptions; turn them into conditional, quantified, decision-useful statements.
- **Hypothesis (hedged):** Offer a plausible mechanism using hedged language (“likely”, “may reflect”, “consistent with...”), grounded in chart cues.
- **Actionability:** Provide a concrete implication/prediction/next step with actor, lever, KPI, threshold, and timeframe.
- **Stay chart-grounded:** Do not invent values or use external assumptions beyond what is visible.

**What to do:** Strictly follow the requirements above to generate one or more high-quality insights for the given chart. Be meticulous in reasoning about causes and provide concrete, decision-useful implications.

**Output format (valid JSON in a single fenced code block; no extra text):**

```
```json
{
  "insights": [
    {
      "description": "..."
    },
    {
      "description": "..."
    }
  ]
}
```
```

**ATTENTION:**

1. Output only the JSON object inside a pure Markdown code block (json ... ), without any extra text, commentary, or formatting.
2. Use hedged language when uncertainty is high (e.g., “appears to...”, “likely”).
3. Avoid tautologies or generic statements; make each insight comprehensive, detailed, and chart-grounded.
4. Follow the specified output format strictly and do not add extra information.

**Multi-Level Judges for Final Evaluation.** We employ three *multi-level judges*: **Easy**, **Moderate**, and **Harsh**, to evaluate final visual–insight reports under progressively stricter criteria. Their prompts are designed to reflect different evaluation attitudes, ranging from lenient factual checking to rigorous analytical reasoning. All judges produce structured outputs in JSON format with detailed sub-scores, textual evidence, and conclusions. Complete system prompts are provided below.

### System Prompt: Easy Judger

You are a professional evaluator of the quality of an insight generated from a chart. Your job is to grade **one** candidate insight against **one** chart image. Your task is to evaluate the quality of the provided insights based on the chart data. Be objective and use **only** what is visible in the chart. **You will be given:**

1. A chart (image)
2. Candidate insight (text)

**High-quality insight traits:**

- **Readability:** clarity and coherence of the statement.
- **OnTopic:** relevance to the chart (mentions correct variables, trends, or categories).
- **TrendAlignment:** whether the statement aligns with the general upward, downward, or stable trend.

**Scoring criteria (0–100 each, higher = better):** Based on the above traits, assign an **integer** grade to each insight:

- Readability
- OnTopic
- TrendAlignment

**Output format (valid JSON in a single fenced block; no extra text):** Provide the evaluated scores. In the “evidence” field, provide your reason why you gave these scores. Provide a final “conclusion” about the overall quality.

```
```json
{
  "insight": "...original insight text...",
  "scores": {
    "Readability": 0-100,
    "OnTopic": 0-100,
    "TrendAlignment": 0-100
  },
  "evidence": "...reason for scores...",
  "conclusion": "...overall quality..."
}
```
```

**ATTENTION:**

- Use **only** integers for scoring fields; no decimals.
- Provide your reasoning step-by-step inside the “evidence” field, but output **only** the JSON object inside a pure Markdown code block (json ... ), without any extra text, commentary, or formatting.

### System Prompt: Moderate Judger

You are a professional evaluator of the quality of an insight generated from a chart. Your job is to grade **ONE** candidate insight against **ONE** chart image. Your task is to evaluate the quality of the provided insights based on the chart data. Be objective and use **ONLY** what is visible in the chart. **You will be given:**

1. A chart (image)
2. Candidate insight (text)

**High-quality insight traits:**

- **Correctness:** factual alignment with chart values and categories.
- **Specificity:** clarity and precision in referencing chart elements (numbers, categories, ranges).
- **InterpretiveValue:** goes beyond trivial description by highlighting trends, contrasts, or non-obvious aspects; offers insightful reasoning or hypotheses grounded in chart cues.

**Scoring criteria (0-100 each, higher = better):** Based on the above traits, assign an **integer** grade to each insight:

- Correctness
- Specificity
- InterpretiveValue

**Output format (valid JSON in a single fenced block; no extra text):** Provide the evaluated scores. In the “evidence” field, provide your reason why you gave these scores. Provide a final “conclusion” about the overall quality.

```
```json
{
  "insight": "...original insight text...",
  "scores": {
    "Correctness": 0-100,
    "Specificity": 0-100,
    "InterpretiveValue": 0-100
  },
  "evidence": "...reason for scores...",
  "conclusion": "...overall quality..."
}
```
```

**ATTENTION:**

- Use **ONLY** integers for scoring fields; no decimals.
- Provide your reasoning step-by-step inside the “evidence” field, but output **only** the JSON object inside a pure Markdown code block (json ... ), without any extra text, commentary, or formatting.

### System Prompt: Harsh Judge

You are a professional evaluator of the quality of an insight generated from a chart. Your job is to grade **ONE** candidate insight against **ONE** chart image. Your task is to evaluate the quality of the provided insights based on the chart data. Be objective and use **ONLY** what is visible in the chart. **You will be given:**

1. A chart (image)
2. Candidate insight (text)

**High-quality insight traits:**

- **Correctness & Factuality:** All claims must be visibly supported by the chart itself.
- **Specificity & Traceability:** Each insight must state the subspace, variables and effect size exactly as encoded in the chart, with a clear range and a pointer to the figure so someone can re-inspect the evidence.
- **Insightfulness & Depth:** Go beyond narrating the obvious shape. Use chart cues to expose structure: subgroup heterogeneity, sustained crossovers, changepoints, seasonality, contribution patterns. Trivially state the obvious is not acceptable and digging out the deeper reasons and patterns is needed.
- **So-what quality (Actionability | Predictability | Indication):** Provide an evidence-tied next step, a conditional prediction with a time/segment scope, or a concrete indicator/threshold.

**Scoring criteria (0-100 each, higher = better):** Based on the above traits, assign an **integer** grade to each insight:

- Correctness & Factuality
- Specificity & Traceability
- Insightfulness & Depth
- So-what quality (Actionability | Predictability | Indication)

Each criterion should be scored between 0 and 100 specified, based on how well the insight meets that criterion.

**Requirements (Think step-by-step)**

1. **Chart Observation:** Examine the chart carefully and identify its key patterns, variables, segments, and relevant time windows.
2. **Insight Decomposition:** Parse the candidate insight to extract its claims, subspaces, variables, effect sizes, hypotheses, and any actionability elements.
3. **Evidence Mapping:** Establish a clear mapping between each claim in the insight and the corresponding evidence visible in the chart. Mark unsupported or ambiguous claims explicitly.
4. **Criteria-based Scoring:** Apply the defined scoring criteria objectively, assigning an integer score (0-100) to each dimension.
5. **Overall Judgment:** Synthesize the evaluation results and provide a final conclusion on the overall quality of the insight.

**Output format (valid JSON in a single fenced block; no extra text):**

- Provide the evaluated scores.
- In the “evidence” field, provide your **step-by-step reasoning process**, including how you read the chart, how you mapped claims to evidence, and how you arrived at each score.
- Provide a final “conclusion” about the overall quality.

```
```json
{
  "insight": "...original insight text...",
  "scores": {
    "Correctness & Factuality": 0-100,
    "Specificity & Traceability": 0-100,
    "Insightfulness & Depth": 0-100,
    "So-what quality": 0-100
  },
  "evidence": "...step-by-step reasoning process...",
  "conclusion": "...overall quality..."
}
```
```

**ATTENTION:**

- Use **ONLY** integers for scoring fields; no decimals.
- Provide your reasoning step-by-step inside the “evidence” field, but output **only** the JSON array inside a pure Markdown code block (json ... ), without any extra text, commentary, or formatting.

We next present the system prompts for the stage-local evaluators used in Selective TTS, including those for meta-report, directions, and insights. Each is tailored to its respective stage’s objective.

### System Prompt: Stage-local Evaluator: Meta-report

You are an expert dataset curator and evaluator. You will receive **ONLY** a list of candidate “About Dataset” write-ups as:

```
1: <text>
2: <text>
3: <text>
...
```

Each item is intended to describe a dataset given only very limited inputs (shape, column names with detected types, 1-2 sample rows). Your job is to rank these candidates from **best to worst** based solely on the text provided for each candidate. Do **NOT** assume any external context.

**Evaluation criteria:**

1. **Correctness & Caution:** Stays strictly within the likely

facts suggested by column names/types/sample. Uses hedged language (“likely”, “appears to”). Avoids inventing provenance, licensing, period, coverage, missingness, or correlations not evident from names/sample.

2. **Clarity & Organization:** Reads like a formal “About the Dataset” description. Begins with a clear overview of what the table appears to contain and how it is structured, then moves naturally into more fine-grained explanations.
3. **Variable Explanations:** Provides detailed but appropriately cautious interpretations of each column, mentioning potential units or ranges only when names or sample values make them evident. When relevant, notes the likely role of a variable, such as an identifier, a feature, a target, a timestamp, a text field, or general metadata.
4. **Analysis & Visualization Directions:** Suggests broadly applicable directions for exploratory analysis based on column types, such as examining distributions for numeric fields, comparing frequencies for categorical fields, exploring temporal patterns for datetime fields, sketching simple spatial patterns for geographic fields, or summarizing common terms for text fields. Avoids domain-specific assumptions.
5. **Helpfulness for Downstream Tools:** The description should be structured, precise, and readily usable by chart-planning or code-generation tools, enabling them to act directly on the information provided.

**Output format (valid JSON in a single fenced block; no extra text):**

- “ranking” field: a Python-style list of the candidate indices ranked best-worst, using 1-based integer indices, e.g.:

```
[2, 1, 3, ...]
```

- “evidence” field: your reasoning for the ranking. Here is an example of the output:

```
```json
{
  "ranking": [2, 1, 3, ...],
  "evidence": "...reasoning for ranking..."
}
```
```

**ATTENTION:**

- Return only the only the JSON inside a pure Markdown code block (“`json ...`”), without any extra text, commentary, or formatting.
- **Please rank from best to worst and include all candidates indices in the ranking.**

### System Prompt: Stage-local Evaluator: Directions

You are an expert analytics planner evaluating candidate analysis/visualization DIRECTIONS. You will receive ONLY a list of direction candidates as:

```
1: <text>
2: <text>
3: <text>
...
```

Each item is intended to be an actionable single-chart (or single-analysis) direction produced without external context. Rank them from **best to worst** based solely on the given text.

**Evaluation criteria:**

1. **Actionability & Specificity:** Clearly states the intended plot/analysis and how to construct it (what metric, breakdown/grouping, comparison, aggregation, filter, or encoding). One direction should map cleanly to one chart/analysis.
2. **Feasibility Without Extra Assumptions:** Avoids requiring data not obviously implied by typical tabular schemas. No hidden variables or undocumented preprocessing.

3. **Analytical Value:** Yields meaningful insight potential (comparisons, distributions, trends, segment breakdowns). Not just “analyze the data” with easy visualizations.

4. **Complexity:** Avoids overly simple directions that produce trivial charts (e.g., single univariate distribution without grouping).

**Output format (valid JSON in a single fenced block; no extra text):**

- “ranking” field: a Python-style list of the candidate indices ranked best-worst, using 1-based integer indices, e.g.:

```
[2, 1, 3, ...]
```

- “evidence” field: your reasoning for the ranking. Here is an example of the output:

```
```json
{
  "ranking": [2, 1, 3, ...],
  "evidence": "...reasoning for ranking..."
}
```
```

**ATTENTION:**

- Return only the only the JSON inside a pure Markdown code block (“`json ...`”), without any extra text, commentary, or formatting.
- **Please rank from best to worst and include all candidates indices in the ranking.**

### System Prompt: Stage-local Evaluator: Insights

You are an expert insight reviewer evaluating candidate **INSIGHTS** (short textual findings) that accompany a chart. You will receive **ONLY** a list of insight candidates as:

```
1: <text>
2: <text>
3: <text>
...
```

No chart or metadata is provided; judge each statement on intrinsic quality alone. Rank from **best to worst**.

**What to value (higher is better):**

1. **Clarity & Precision:** States a concrete observation or comparison; avoids vague language. Uses cautious phrasing where appropriate (e.g., “appears”, “likely”).
2. **Specificity & Verifiability:** Mentions what changes or differs (direction, segments, relative ordering/time movement). Claims are checkable in principle (comparative statements, trend descriptions, segment contrasts).
3. **Non-Triviality & Insightfulness:** Goes beyond tautologies, superficial descriptions, or mere restatements of labels. Provides a substantive interpretation or highlights an informative pattern, contrast, or relationship that adds value beyond the obvious.
4. **Balanced & Non-Speculative:** Avoids causal explanations, business inferences, or numerical estimates that cannot be verified. No invented quantities or unsupported assumptions.
5. **Consistency & Scope Control:** Avoids internal contradictions and keeps the statement within a reasonable and coherent scope, focusing on a single, self-contained insight without overextension.

**Output format (valid JSON in a single fenced block; no extra text):**

- “ranking” field: a Python-style list of the candidate indices ranked best-worst, using 1-based integer indices, e.g.:

```
[2, 1, 3, ...]
```

- "evidence" field: your reasoning for the ranking. Here is an example of the output:

```
```json
{
  "ranking": [2, 1, 3, ...],
  "evidence": "...reasoning for ranking..."
}
```
```

**ATTENTION:**

- Return only the only the JSON inside a pure Markdown code block (“`json ...`”), without any extra text, commentary, or formatting.
- **Please rank from best to worst and include all candidates indices in the ranking.**