# Towards 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), iterative refinement guided by reward signals. However, many real-world tasks involve multi-stage pipeline whose final outcomes lack verifiable rewards or sufficient data to train robust reward models, making judge-based refinement prone to accumulate error over stages. We propose Selective TTS, a *process-based refinement* framework that scales inference across different stages in multi-agent pipeline, instead of repeated refinement over time by 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 the data science pipeline, we build an end-to-end multi-agent pipeline for generating visually insightful charts and report of given dataset, and design a reliable LLM-based judge model, aligned with human experts (Kendall's $\tau$=0.55). Our proposed selective TTS then improves insight quality under a fixed compute budget, increasing mean scores from 61.64 to 65.99 while reducing variance. We hope our findings serve as the first step toward to scaling complex, open-ended tasks with unverifiable rewards, such as scientific discovery and story generation. Our code and generated reports are publicly available [1]

## 1 Introduction

LLM agents increasingly improve performance by allocating more inference-time compute to *branch, reflect, and verify* before committing to answers. Approaches such as Self-Consistency (Wang et al., 2023), Tree-of-Thoughts (Yao et al., 2023), and evolutionary refinement like AlphaEvolve (Novikov et al., 2025) show that iterative reasoning and resampling at test time can scale solution quality comparably to model or training-time gains (Snell et al., 2024; Liu et al., 2025b).

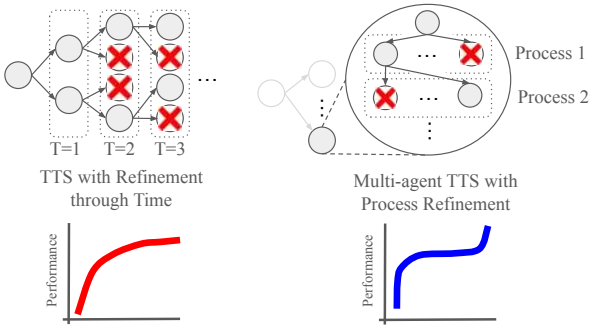



Figure 1: Comparison between traditional TTS through refinement over time (left) (Novikov et al., 2025), and our proposed process refinement in multi-agent pipeline (right). They show different styles of scaling behaviors.

Many complex, real-world tasks involve multi-stage pipelines, such as analysis, generation, evaluation, and refinement, where the final outcomes are *unverifiable* (e.g., insightfulness of generated charts, engagingness of generated story). In most cases, these unverifiable objectives lack of sufficient training data to train robust reward models due to the difficulty of defining the problem and underlying principles to define the problem tasks. In such cases, practitioners often rely on LLM-as-Judge feedback (Zheng et al., 2023), yet repeated refinement using judge-generated scores tends to accumulate and amplify early misjudgments. Consequently, traditional time-based TTS can drift away from genuinely human-preferred outcomes, degrading rather than improving quality.

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 outputs through time, Selective TTS allocates compute across pipeline stages and applies stage-specific evaluators to prune low-quality branches early.

[1] https://minnesotanlp.github.io/insight-scaling-webpage

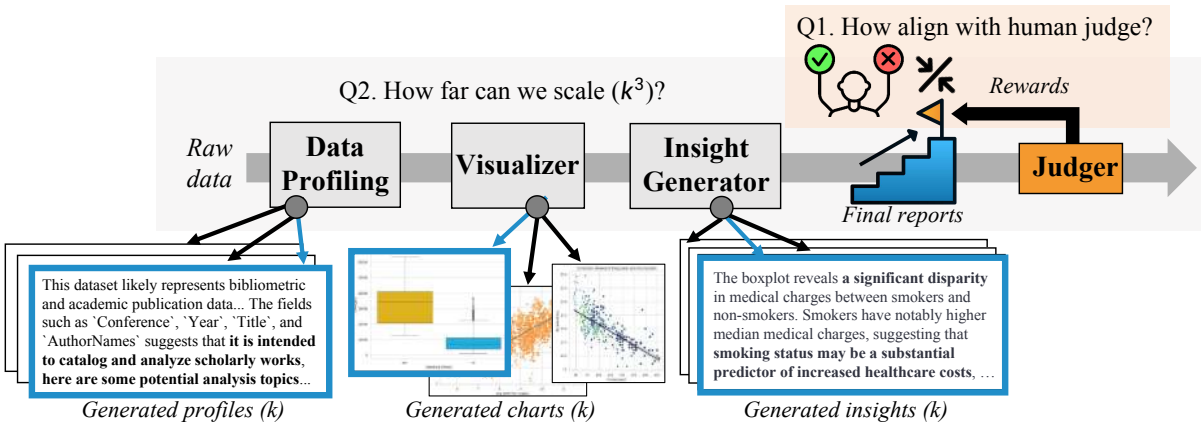



Figure 2: Our multi-agent data analysis pipeline aims to answer two key questions: (RQ1) whether judge-guided scaling can align with human experts, and (RQ2) how much performance can be further scaled within the same compute budget using the proposed selective TTS.

This design confines each judge's influence to its local stage, reducing error propagation and yielding more stable, interpretable scaling behavior (Fig. 1). While traditional refinement accumulates noise over time, Selective TTS redistributes scaling horizontally across independent pipeline executions to achieve consistent quality improvements under fixed compute.

Data analysis transforms raw data into visualizations and then into insight driven reports. Each stage, from choosing what to analyze to designing and interpreting charts, requires substantial expertise and intuition, making it difficult to evaluate the final report with a single scalar reward because the notion of a "good" insight is often subjective and depends heavily on domain knowledge and analytical experience (Wang et al., 2025b). Most prior work on automating data analysis like Data Science Agents (Google Labs) fall into following rigid linear scripts and therefore plotting simple charts of surface-level outputs.

We take this inherently challenging and unverifiable process as our case study. This setting provides a grounded environment for systematically examining how *process-based refinements* during test time can improve overall 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 generated report quality, and identify a pseudo–ground-truth judge that best matches human expert judgment ($\tau$=0.55), providing a human-aligned reference for subsequent experiments.
- We propose and evaluate *Selective TTS*, a pruning-based compute allocation strategy across pipeline stages, that improves insight quality of reports (+4.4 gain, –31% variance) under the same compute budget.

Together, 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 *Judger 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 work by Gan et al.

(2025) but simplified for the sake of efficient scaling. All examples shown in §2.1 are illustrative outputs generated using the pipeline with `GPT-4o` instantiated as the LLM backbone on the VIS publication dataset (Isenberg et al., 2017). This use is solely for producing clear examples for exposition and does not influence any experimental results reported in later sections. Detailed explanation of the pipeline design is in Appendix §H

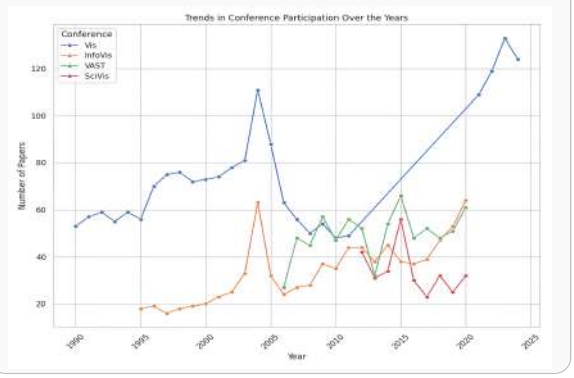
**(1) Data Profiling.** The *Data Profiling* stage summarizes key metadata of the dataset (shape, schema, inferred types, small samples) and emits a concise *metadata report* with plausible analysis directions. This compact contract guides downstream visualization (valid columns, sensible encoding),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

An example generated metadata report

This dataset appears to represent a bibliographic collection of conference papers, likely covering scientific or technical domains. It contains a mixture of numeric, categorical, and textual attributes ... Such data can support analyses of research trends, citation patterns, and scholarly impact. Here are several potential directions ...

**(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 judger. The result is a vetted set of figures ready for interpretation. (see an example plot generated below from the previous profile).

An example generated chart




**(3) Insight Generation.** For each vetted chart, the *Insight Generation* stage samples textual insights under specific instructions that emphasizes (i) observation fidelity and evidential completeness, (ii) traceability to specific chart elements, (iii) nontriviality and novelty, and (iv) actionability. An example insight corresponding to the above figure is shown below.

An example generated insight

Between 2005 and 2010, both the Vis and InfoVis conferences experienced a clear decline in the number of accepted papers, with Vis decreasing from around 110 papers in 2005 to roughly 60 in 2010, while InfoVis dropped from about 60 to around 40 during the same period. ... This shift may reflect the field's evolving structure, ... Understanding these dynamics can help ...

**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 *Judger* 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 judgers, 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 by the same judger *three* independent times, 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 Judgers** 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 judgers may exhibit systematic biases or preference shifts, making it unclear whether judge-guided improvements after scaling genuinely reflect human-perceived quality.

To ensure that judge-guided scaling aligns with human expert preferences, we explicitly validate and select a judger whose rankings are consistent with human judgments. We therefore conduct a human evaluation using pairwise preference rank-

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

Table 1: Prompt snippets for the three judgers. 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.2.

ings over sampled final reports, and measure the agreement between human rankings and each candidate judger. This process allows us to identify a judger that reliably reflects human quality assessments, which is then used as the proxy signal for evaluating report quality under test-time scaling.

### 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, we now test how far we can scale insight quality under a fixed compute budget (i.e., finding the best outcome out of a $k^3$ combination of $k$ profiles, $k$ charts, and $k$ reports in Figure 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 (Figure 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 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

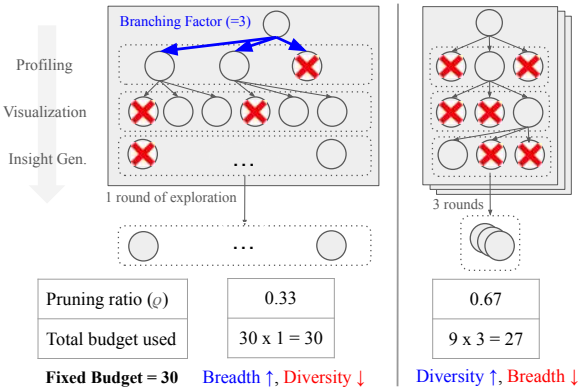



Figure 3: Overview of **Selective Test-Time Scaling** under a fixed compute budget (e.g., 30). Stage-specific evaluators perform pruning across the multi-agent pipeline while controlling the branching factor at each stage. Higher pruning ratios ($\rho$) promote more rounds of exploration and diversity across runs, while reducing within-round breadth. Details on budget accounting and branching control appear in §2.2.2.

to be allocated to high-potential branches.

**Pruning Schedule.** Let $b_s$ denote the branching factor at stage $s \in \{\text{metadata reports}, \text{directions}, \text{insights}\}$. Given a pruning ratio $\rho$, we retain

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

candidates to pass forward.[2] 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 judger tailored to its

[2]We enforce $\max(1, \cdot)$ to avoid pathological collapse at high pruning rates.

stage: a metadata judger ranks metadata reports, a visualization judger ranks directions of visual charts, and an insight judger ranks drafted insights. This process-level allocation reduces the accumulation of judging noise across time and yields more stable scaling behavior under fixed budget.

#### 2.2.2 Budget Accounting and Control

To ensure fair comparison, we hold the total compute constant across all pruning configurations Compute is measured by the *number of LLM calls* (generation + stage-local judging + final evaluation) across all agents. Since all runs share the same model and environment, this call-based measure reliably tracks inference cost without requiring token-level accounting. For each run, the budget is computed as follows:

- *Profiling:* $+1$ call for metadata reports generation, and $+1$ for pruning (if applied).
- *Visualization:* for each metadata report (Profile), $+1$ for visualization direction, and $+1$ call for pruning (if applied).
- *Chart Rendering:* $+2$ calls per visualization direction (code generation + 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

Given a baseline budget $B$ ($\rho$=0), we adjust the number of runs such that the total LLM calls under each pruning ratio $\rho$ closely matches $B$. Our implementation dynamically monitors the accumulated budget during execution and incrementally supplements additional runs until the target budget is reached; the detailed procedure is described in Appendix §E.2. This normalization ensures that improvements observed under Selective TTS reflect more effective compute allocation rather than increased compute. We further validate in §4.4 that LLM-call–based budgeting is a reliable proxy for token-level budgets and exhibits consistent performance trends.

#### 2.2.3 Budget Complexity

The run-level budget complexity can be approximated as:

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

where $|V|$ is the number of verified charts and $n_s' = \max(1, \lceil (1-\rho) b_s \rceil)$ denotes the surviving branches after pruning at stage $s$. The first term reflects generation and evaluation costs with quadratic dependence on branch size, while the second captures the marginal overhead from pruning. A full derivation appears in Appendix §E.1.

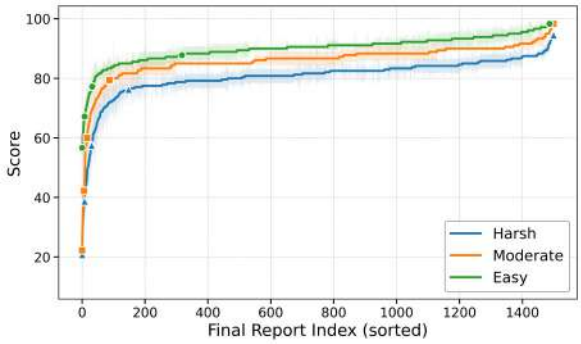



Figure 4: Sorted overall score curves under three judgers (easy, moderate, harsh) on VIS Publication dataset. See Appendix §C for the Medical Insurance dataset.

## 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).

### 3.1 Small-Sized Human Alignment Experiment

**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 judger. 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 shows an example sorted score curves by three judgers 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 pref-

| Judger | VIS Publication Dataset | | | Medical Insurance Dataset | | |
|---|---|---|---|---|---|---|
| | Kendall's $\tau$ (↑) | Spearman's $\rho$ (↑) | Kendall's $W$ (↑) | Kendall's $\tau$ (↑) | Spearman's $\rho$ (↑) | Kendall's $W$ (↑) |
| Easy | 0.40±0.24 | 0.53±0.24 | **0.64** | 0.55±0.30 | 0.62±0.26 | 0.54 |
| Moderate | 0.45±0.17 | 0.55±0.23 | 0.51 | 0.40±0.00 | 0.55±0.08 | **0.65** |
| **Harsh** | **0.55±0.30** | **0.60±0.37** | 0.59 | **0.65±0.30** | **0.72±0.27** | **0.64** |

Table 2: Alignment metrics (mean ± std across 4 human annotators) between each judger and human annotations on two datasets. Higher values indicate stronger agreement (1 denotes perfect correlation).

erences 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 judger'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 Appendix §D.1. We then identify the pseudo ground-truth judger $\tilde{j}$ by selecting the judger with the strongest agreement with human rankings:

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

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

**Report Generation and Judging Configuration.** With `Qwen-2.5-VL-32B-Instruct` as the generation backbone, we run the full pipeline with test-time branching to produce *final reports*, each containing a visualization paired with its corresponding textual insight. To ensure that our judger design is robust, we use `GPT-4o` as a strong judging backbone throughout the human-alignment study, which helps minimize annotation bias arising from model-level limitations. For stability, each report is evaluated by the same judger three independent times, and the final score is obtained by averaging these repeated evaluations (majority-voting). We evaluate two representative tabular datasets—*VIS Publication Data* (Isenberg et al., 2017) and *Medical Insurance* (Abdelghany, 2021)[3]—each yielding approximately 1,500 final reports. From these results, we obtain the **sorted index curves** of report scores, which visualize the score distribution across pruning ratios. These curves then serve as the basis for score-based sampling. Detailed dataset information is provided in Appendix §B.

[3]We use two diverse, multi-attribute tabular datasets to enable varied charts/insights and to test cross-domain generality.

### 3.2 Large-Scale 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 judger) 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 cross-model variation, decoding parameter choices, and dataset generalization 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

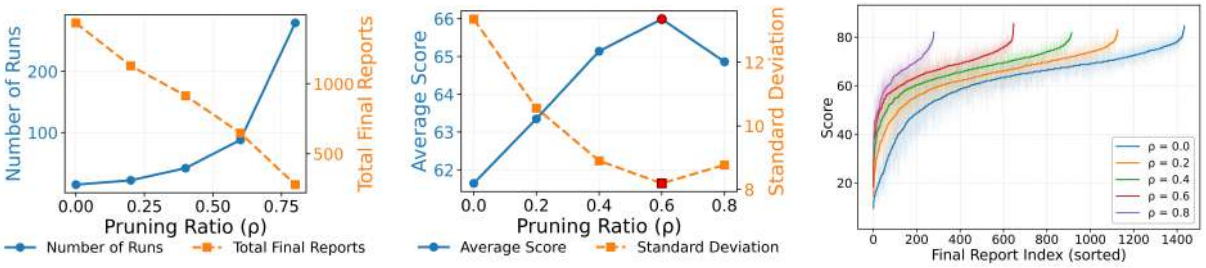
We like to answer the two key research questions:

**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)

### 4.1 Judge Selection via Human Alignment

Table 2 shows that the **harsh** judger best aligns with human preferences across both datasets (highest $\tau$




(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.

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-local 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 judger scores, is provided in Appendix §D.4. Further details on the annotation interface, protocol, and rater calibration procedure can be found in Appendix §D.2 and Appendix §D.3.

### 4.2 Selective Pruning for Scaling

We test whether Selective TTS improves overall report quality under a fixed compute budget. Figures 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. 3), 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 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 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.** Mean scores increase steadily from $\rho = 0$ to $\rho = 0.6$ (from 61.64 to 65.99), accompanied by a clear reduction in variance (from 13.36 to 8.19). 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 judger, 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 3.

### 4.3 Ablation Study

To clarify the role of stage local pruning in Selective TTS, we conduct two ablation studies: replacing 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. Detailed are reported in Appendix §E.4.

**Random Sampling.** In this setting, we remove all stage-local evaluators and sample candidates at each stage purely at random according to the

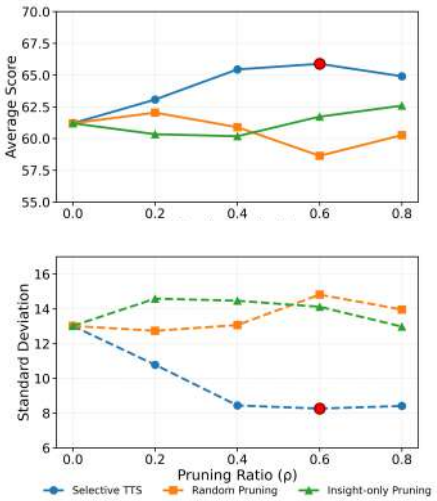
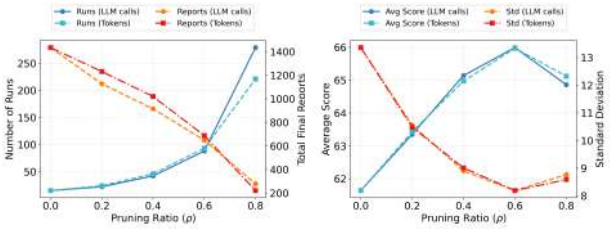



Figure 6: Ablation results comparing Selective TTS with Random Pruning and Final Stage only (Insight-Only) Pruning across varying pruning ratios $\rho$: (a) Average scores and (b) Standard deviation




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

Figure 7: Effects of Selective TTS with LLM-calls as budget and Token budget

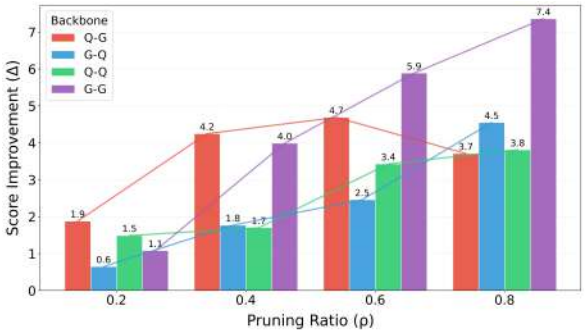



Figure 8: Score improvements ($\Delta$) achieved by Selective TTS across four generator–judger backbone combinations (Q→G, G→Q, Q→Q, and G→G).

pruning ratio $\rho$. Based on Figures 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.** We analyze whether the gains arise solely from ranking at the final stage. In this setting, pruning is applied only during insight generation, while all earlier stages keep full branching. This isolates the effect of downstream ranking, since the final judge evaluates chart and insight pairs. 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 becomes 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

To evaluate the reliability and generality of Selective TTS, we perform additional analyses: comparing LLM call and token level budget accounting, examining cross-model variation, analyzing sensitivity to decoding parameters such as temperature, evaluating transfer to a different dataset, and testing robustness to output length variation. Detailed results are provided in Appendix §E.5.

**Token-Level vs. LLM-Call Budget.** To ensure that using LLM calls as the compute budget does not distort scaling behavior, we rerun the full experiment while controlling the number of output tokens instead of model invocations. As shown in Fig. 7, performance curves under both budgeting schemes are highly similar. The number of runs and total generated tokens exhibit nearly identical trends across pruning ratios, indicating comparable exploration exploitation trade offs. Since the qualitative behavior of Selective TTS is preserved under token level budgeting, LLM call budgets are a reliable proxy. In addition, LLM call budgeting is easier to control and reproduce, as token counts vary with prompt verbosity, chart complexity, and model specific generation behavior, whereas model calls provide a stable and environment agnostic compute unit.

**Cross-Model Variation.** To assess sensitivity to the model backbone, we evaluate four generator judger pairings across two model families: Q→Q, Q→G, G→Q, and G→G, where Q denotes

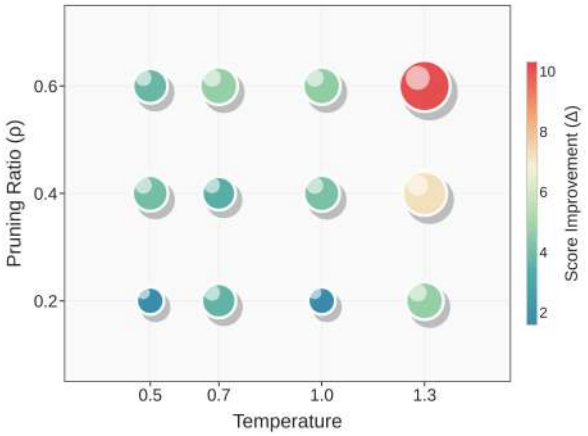



Figure 9: Score improvements of Selective TTS across different decoding temperatures and pruning ratios $\rho$. Each bubble represents the improvement over the baseline ($\rho$=0), with color and size indicating the magnitude of the gain.

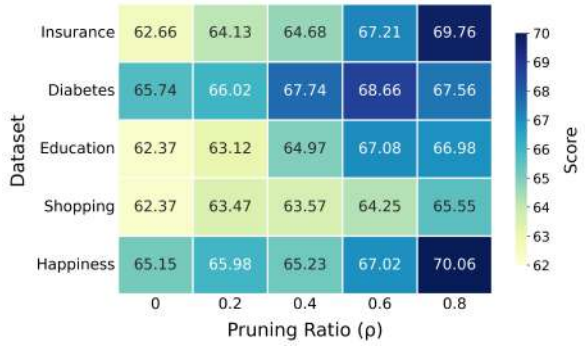



Figure 10: Score heatmap across datasets and pruning ratios $\rho$. Selective TTS shows consistent performance gains across all datasets.

`Qwen-2.5-VL-32B` and G denotes `GPT-4.1-nano`. For each pairing, we run the baseline system at $\rho = 0$ for eight runs to estimate a matched compute budget, then apply Selective TTS at higher pruning ratios. As shown in Fig. 8, all configurations show consistent improvements as $\rho$ increases. While gain magnitudes vary slightly across backbones, stronger pruning reliably improves quality in every case, indicating that Selective TTS generalizes across model families and is not driven by family specific biases.

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

To test sensitivity to decoding choices, we vary the generator temperature while fixing top-$p$, and

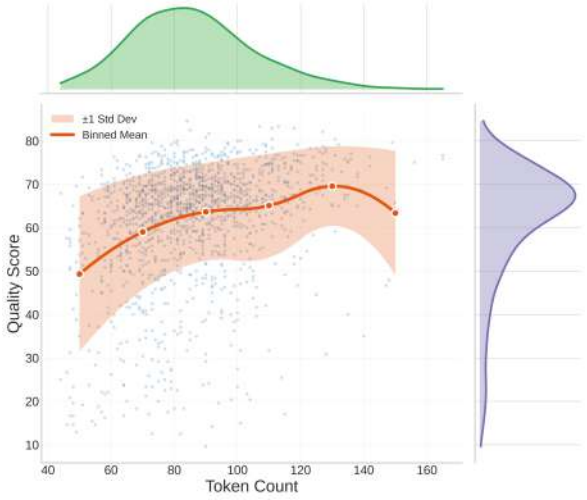



Figure 11: Insight length vs quality score.

apply Selective TTS with the same pruning ratios. For each temperature, we first run the baseline system ($\rho = 0$) for eight runs to estimate a matched compute budget, then apply Selective TTS at higher pruning ratios under this fixed budget. As shown in Fig. 9, Selective TTS consistently outperforms the baseline across all temperatures, even as generation becomes more deterministic at lower values. The modest dip around 0.4 to 0.6 likely reflects reduced diversity, which weakens stage local discrimination. Nonetheless, higher pruning ratios continue to yield larger gains, and the overall improvement pattern remains stable. Extremely high temperatures introduce execution errors and reduce usable runs, as detailed in Appendix §E.5 Table 9.

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

**Generalization Across Datasets.** To test generalization beyond the VIS dataset, we evaluate Selective TTS on five additional tabular datasets spanning diverse semantic and statistical characteristics: *Medical Insurance* (Abdelghany, 2021), *Diabetes* (Khare, 2020), *Education Performance* (Fatima, 2023), *Customer Shopping Behavior* (Siddiqa, 2022), and the *World Happiness Report* (Buttar, 2023). Dataset details are provided in Appendix §B. For each dataset, we run the baseline system ($\rho = 0$) eight times to estimate a matched compute budget, then apply Selective TTS at higher pruning ratios.

As shown in Fig. 10, all datasets show consistent score improvements as the pruning ratio

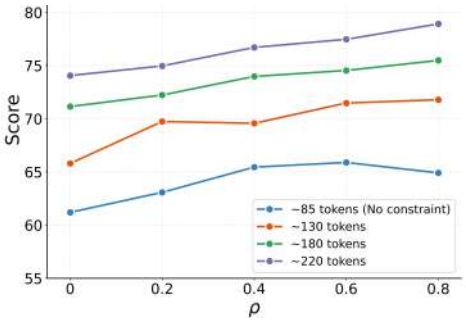



Figure 12: Scaling behavior of Selective TTS under different insight-length regimes.

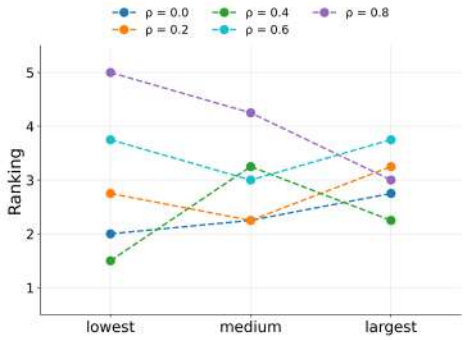



Figure 13: Mean annotator rankings across pruning ratios. Rankings converge as high-scoring report become comparably strong. (higher is better)

$\rho$ increases, indicating that the scaling behavior is not dataset specific. Moderate pruning ($\rho \in 0.2, 0.4, 0.6$) yields steady gains across all datasets, while the most aggressive setting ($\rho = 0.8$) produces mixed results, with weaker improvements on some datasets. This suggests that excessive pruning can over eliminate promising branches depending on dataset complexity and noise. Overall, Selective TTS generalizes well across heterogeneous domains, with moderate pruning offering the most reliable trade off between exploration and pruning depth.

**Sensitivity to Insight Length.** Under the baseline setting ($\rho = 0$), we observe a positive correlation between insight length and evaluation score: longer insights generally receive higher scores, likely because they provide richer context, clearer reasoning, and more supporting detail. Figure 11 illustrates this trend at the instance level.

Motivated by this, we test whether Selective TTS continues to yield gains as output length increases, and whether pruning remains effective beyond regimes already favored by the evaluator. To separate length effects from scaling behavior, we group insights into coarse token length regimes and evaluate Selective TTS within each group. We enforce length variation through prompt level constraints and, for each regime, run the baseline system ($\rho = 0$) for eight runs to estimate the average output length. As shown in Fig. 12, increasing the pruning ratio $\rho$ consistently improves performance across all length regimes, with highly similar scaling trends. Meanwhile, baseline scores rise with output length, confirming that longer insights are generally rated more highly.

### 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. 14), the lowest scoring report contains only surface level keywords, the mid scoring report summarizes overall publication trends, and the highest scoring report explains why certain outliers 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, 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 12 in Appendix §F. This trend is further illustrated in Figure 13, 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 SelectiveTTS 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.

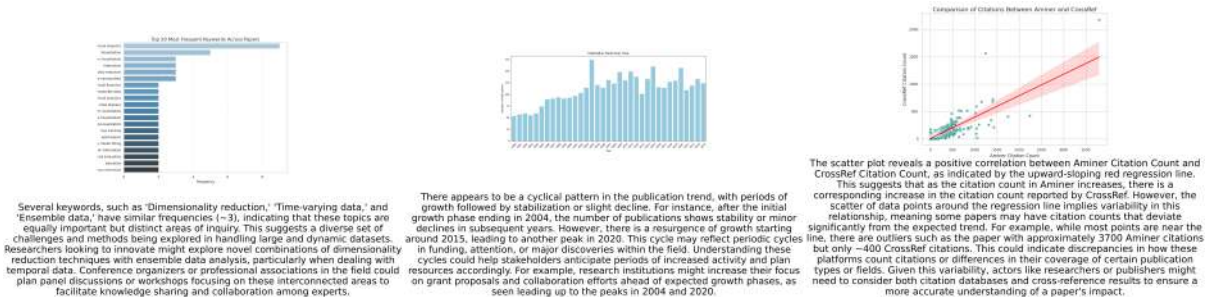



Figure 14: Reports for $\rho = 0.6$ at the lowest (32.92), median (58.3), and highest scores (84.2).

## 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.

**LLM Agents for Data Science and Visual Insight Generation.** LLM-based agents increasingly automate data-centric workflows such as 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 largely 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 treat 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 judger 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 capture the diversity of real-world data science tasks. Second, although we explored larger budgets and an alternate dataset, the overall compute and dataset scales remain moderate. Future work should expand dataset coverage, model diversity, and adaptive parameter tuning to better characterize selective scaling behavior in multi-agent pipelines.

**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 rea-

soning. We recommend that such systems be used to assist, rather than replace, human analysts, with proper calibration and 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.

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. In *Proceedings of the IEEE VIS Workshop on Visual Analytics and Generative AI (VIS x GenAI)*, Vienna, Austria. IEEE.

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. *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 Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, 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 Tailor, 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, Ngân 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 Siddiqa. 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* (Siddiqa, 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.

**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.

**Happiness.** 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.

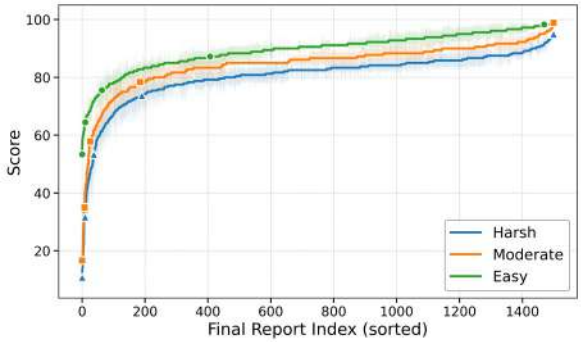



Figure 15: Sorted overall score curves under three judgers (easy, moderate, harsh) on Medical Insurance dataset.

**Student Performance.** The Student Performance dataset contains structured records of secondary-school students, including demographic attributes, family background variables, study habits, and academic outcomes. Features include age, gender, parental education, study time, absences, and numerical exam grades, combining categorical descriptors with performance metrics.

**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.** Figure 15 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 human annotation study. This section outlines the overall evaluation framework, including (1) the rank-based correlation metrics used to quantify alignment, (2) the annotation interface and protocol for pairwise comparison, (3) the calibration procedures ensuring annotation consistency, and (4)

| Setting | Runs | Total Final Reports | Score (Avg. ± Std.) | Gen. Budget | Prune Budget |
|---|---|---|---|---|---|
| Baseline ($\rho = 0$) | 15 | 1435 | 61.64 ± 13.36 | 2507 | 0 |
| $\rho = 0.2$ | 22 | 1128 | 63.35 ± 10.56 | 2152 | 384 |
| $\rho = 0.4$ | 42 | 915 | 65.14 ± 8.90 | 2046 | 459 |
| $\rho = 0.6$ | 88 | 648 | **65.99 ± 8.19** | 1920 | 284 |
| $\rho = 0.8$ | 279 | 279 | 64.86 ± 8.77 | 1674 | 837 |

Table 3: Main results under different pruning ratios $\rho$ on the VIS dataset. Selective TTS progressively reduces the number of final reports while increasing average quality (higher mean, lower variance) under nearly fixed compute budget.

illustrative examples comparing human and model preferences.

### D.1 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 $R_{\cdot i}$ denote the sum of ranks assigned to item $i$, and let $\bar{R}$ be their mean. Kendall's $W$ is defined as:

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

$W$ 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 Judger aligns with human preferences.

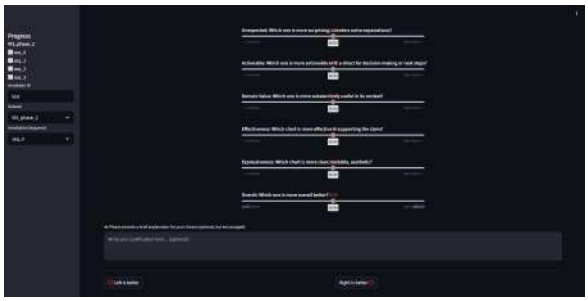
**Interpretation.** Kendall's $\tau$ emphasizes fine-grained pairwise ordering consistency; Spearman's $\rho$ captures global monotonic structure and penalizes large rank deviations; Kendall's $W$ summarizes multi-annotator agreement. Together, these metrics offer complementary perspectives on how closely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

### D.2 Annotation Interface and Protocol

Figure 16 illustrates the human annotation interface used in our study. The interface consists of two main sections: the **upper panel** (Figure 16a) and the **lower panel** (Figure 16b).

**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.


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

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.3 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.

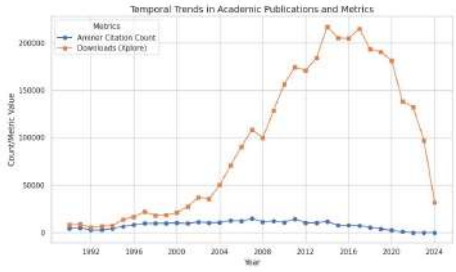
All annotators were *graduate students with prior experience in data visualization and analytic reasoning*, ensuring familiarity with interpreting charts and evaluating insights.

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.4 Example of Human–Judger Preference Comparison

Figure 17 illustrates one representative pairwise comparison from the human annotation interface, showing two visual–insight reports displayed side by side along with their corresponding judger 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 judger scores were *not visible* to annotators; they were only added here for post-hoc comparison and visualization purposes.



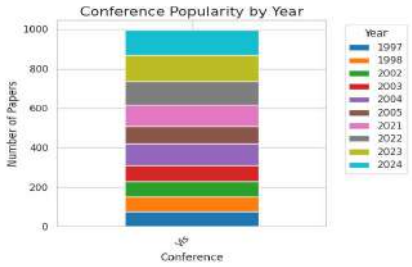

The dramatic decline in academic paper downloads (Xplore) starting around 2020 and continuing to 2024 suggests a significant shift in how researchers access scholarly content. This trend may reflect a growing preference for alternative platforms or open-access resources, possibly due to increased awareness 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.

**Annotation 1: Left one is better**

**Annotation 2: Left one is better**

**Annotation 3: Left one is better**

**Annotation 4: Right one is better**




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.

Figure 17: Representative example of a human–judger preference comparison.

# 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 $s \in \{\text{meta-reports}, \text{directions}, \text{insights}\}$, let $b_s$ denote its branching factor, and $n'_s = \max(1, \lceil (1-\rho) b_s \rceil)$ the number of retained candidates after pruning. The total budget per run, $B_{\text{run}}(\rho)$, can then be decomposed into the sum of all major LLM calls throughout the pipeline:

$$B_{\text{run}}(\rho) = \big[\, 1 + \mathbb{I}[\rho > 0] + n'_s\big(1 + \mathbb{I}[\rho > 0] + 2n'_s + |V| + |V|\mathbb{I}[\rho > 0] + |V| n'_s\big) \,\big].$$

Each term corresponds to a specific operation:

- $1 + \mathbb{I}[\rho > 0]$: metadata report generation and pruning;
- $1 + \mathbb{I}[\rho > 0]$ (inside brackets): direction generation and pruning;
- $2n'_s$: chart code generation and verification;
- $|V|$: insight generation;
- $|V|\mathbb{I}[\rho > 0]$: insight pruning;
- $|V| n'_s$: insight evaluation by the stage-specific judger.

Expanding and regrouping terms by order, we obtain the intermediate expression:

$$B_{\text{run}}(\rho) = \underbrace{\Big((2 + |V|){n'_s}^2 + n'_s|V| + n'_s + 1\Big)}_{\text{Generation Budget}} + \underbrace{\mathbb{I}[\rho > 0]\Big(n'_s|V| + n'_s + 1\Big)}_{\text{Pruning Budget}}.$$

By grouping dominant terms and expressing in asymptotic form, we obtain:

$$B_{\text{run}}(\rho) = \mathcal{O}\big({n'_s}^2|V| + n'_s|V| + n'_s\big) + \mathcal{O}\big(\mathbb{I}[\rho > 0]\,(n'_s|V| + n'_s)\big).$$

Since the quadratic and linear terms with respect to $n'_s$ and $|V|$ dominate the total cost, and lower-order constants contribute negligibly, we can further simplify the expression to:

$$B_{\text{run}}(\rho) \approx \mathcal{O}(n'^2_s|V| + n'_s|V|) + \mathcal{O}(\mathbb{I}[\rho > 0]\, n'_s|V|).$$

### 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 $n^\star_\rho$ such that $B_\rho^{(n^\star_\rho)}$ 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.[4] 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 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:

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

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 $B_\rho^{(n^\star_\rho)}$ 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. Figure 7).

### E.3 Detailed Results for Main Experiments

Table 3 reports the full quantitative results for Selective TTS under different pruning ratios $\rho$ on the VIS dataset. These values correspond to the performance trends visualized in Figure 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 65.99), accompanied by a substantial reduction in variance (from 13.36 to 8.19), 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

Figure 6 visualizes the comparative behavior of Selective TTS, Random Pruning, and Insight-only Pruning across pruning ratios $\rho$. Tables 4, 5, and 6 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.

[4] In practice, this can be as simple as running a few preliminary runs and taking their empirical statistics.

| Setting | Runs | Total Final Reports | Score (Avg. ± Std.) | Gen. Budget | Prune Budget |
|---|---|---|---|---|---|
| Baseline ($\rho = 0$) | 8 | 805 | 61.20 ± 13.02 | 1403 | 0 |
| $\rho = 0.2$ | 12 | 644 | 63.08 ± 10.78 | 1222 | 218 |
| $\rho = 0.4$ | 25 | 522 | 65.45 ± 8.44 | 1169 | 263 |
| $\rho = 0.6$ | 50 | 364 | **65.89 ± 8.26** | 1081 | 329 |
| $\rho = 0.8$ | 156 | 156 | 64.91 ± 8.41 | 936 | 468 |

Table 4: Selective TTS under different pruning ratios $\rho$ (small-scale setting).

| Setting | Runs | Total Final Reports | Score (Avg. ± Std.) | Gen. Budget | Prune Budget |
|---|---|---|---|---|---|
| Baseline ($\rho = 0$) | 8 | 805 | 61.20 ± 13.02 | 1403 | 0 |
| $\rho = 0.2$ | 16 | 696 | 62.05 ± 12.74 | 1354 | 0 |
| $\rho = 0.4$ | 29 | 597 | 60.89 ± 13.07 | 1385 | 0 |
| $\rho = 0.6$ | 70 | 452 | 58.64 ± 14.82 | 1408 | 0 |
| $\rho = 0.8$ | 234 | 234 | 60.27 ± 13.96 | 1404 | 0 |

Table 5: Random pruning baseline under different pruning ratios $\rho$ (small-scale setting).

| Setting | Runs | Total Final Reports | Score (Avg. ± Std.) | Gen. Budget | Prune Budget |
|---|---|---|---|---|---|
| Baseline ($\rho = 0$) | 8 | 805 | 61.20 ± 13.02 | 1403 | 0 |
| $\rho = 0.2$ | 9 | 608 | 60.34 ± 14.59 | 1187 | 152 |
| $\rho = 0.4$ | 13 | 564 | 60.18 ± 14.47 | 1271 | 188 |
| $\rho = 0.6$ | 12 | 410 | 61.73 ± 14.12 | 1177 | 205 |
| $\rho = 0.8$ | 14 | 244 | 62.60 ± 12.98 | 1151 | 244 |

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

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

Table 7: Robustness to token-level budgeting: results under matched output-token budgets.

| Setting | Runs | Total Final Reports | Score (Avg. ± Std.) | Gen. Budget | Prune Budget |
|---|---|---|---|---|---|
| **Q–G Backbone** | | | | | |
| Baseline ($\rho = 0$) | 8 | 805 | 61.20 ± 13.02 | 1403 | 0 |
| $\rho = 0.2$ | 12 | 644 | 63.08 ± 10.78 | 1222 | 218 |
| $\rho = 0.4$ | 25 | 522 | 65.45 ± 8.44 | 1169 | 263 |
| $\rho = 0.6$ | 50 | 364 | 65.89 ± 8.26 | 1081 | 329 |
| $\rho = 0.8$ | 156 | 156 | 64.91 ± 8.41 | 936 | 468 |
| **G–Q Backbone** | | | | | |
| Baseline ($\rho = 0$) | 8 | 788 | 70.18 ± 6.87 | 1372 | 0 |
| $\rho = 0.2$ | 12 | 586 | 70.83 ± 6.19 | 1132 | 202 |
| $\rho = 0.4$ | 23 | 474 | 71.95 ± 5.91 | 1110 | 246 |
| $\rho = 0.6$ | 53 | 338 | 72.64 ± 6.55 | 1056 | 322 |
| $\rho = 0.8$ | 152 | 152 | 74.73 ± 5.71 | 912 | 456 |
| **Q–Q Backbone** | | | | | |
| Baseline ($\rho = 0$) | 8 | 650 | 71.53 ± 6.08 | 1140 | 0 |
| $\rho = 0.2$ | 10 | 476 | 73.02 ± 6.24 | 929 | 165 |
| $\rho = 0.4$ | 21 | 396 | 73.23 ± 5.63 | 927 | 207 |
| $\rho = 0.6$ | 44 | 280 | 74.96 ± 5.73 | 874 | 266 |
| $\rho = 0.8$ | 127 | 127 | 75.34 ± 5.69 | 762 | 381 |
| **G–G Backbone** | | | | | |
| Baseline ($\rho = 0$) | 8 | 747 | 59.68 ± 13.67 | 1312 | 0 |
| $\rho = 0.2$ | 11 | 590 | 60.76 ± 13.74 | 1127 | 201 |
| $\rho = 0.4$ | 20 | 491 | 63.67 ± 10.91 | 1096 | 245 |
| $\rho = 0.6$ | 45 | 334 | 65.57 ± 8.93 | 997 | 303 |
| $\rho = 0.8$ | 146 | 146 | 67.04 ± 8.27 | 876 | 438 |

Table 8: Cross-model robustness: generation–judge backbone combinations (Q–G, G–Q, Q–Q, G–G).

| Setting | Runs | Total Final Reports | Score (Avg. ± Std.) | Gen. Budget | Prune Budget |
|---|---|---|---|---|---|
| **Temperature = 0.5** | | | | | |
| Baseline ($\rho = 0$) | 8 | 675 | 61.38 ± 14.10 | 1192 | 0 |
| $\rho = 0.2$ | 11 | 556 | 63.27 ± 12.34 | 1057 | 189 |
| $\rho = 0.4$ | 19 | 456 | 65.51 ± 7.94 | 991 | 223 |
| $\rho = 0.6$ | 41 | 312 | 65.26 ± 9.03 | 914 | 278 |
| **Temperature = 0.7** | | | | | |
| Baseline ($\rho = 0$) | 8 | 690 | 60.71 ± 13.61 | 1221 | 0 |
| $\rho = 0.2$ | 12 | 560 | 64.46 ± 9.88 | 1072 | 192 |
| $\rho = 0.4$ | 20 | 444 | 64.24 ± 9.92 | 990 | 222 |
| $\rho = 0.6$ | 43 | 310 | 65.52 ± 8.67 | 928 | 282 |
| **Temperature = 1.0** | | | | | |
| Baseline ($\rho = 0$) | 8 | 805 | 61.20 ± 13.02 | 1403 | 0 |
| $\rho = 0.2$ | 12 | 644 | 63.08 ± 10.78 | 1222 | 218 |
| $\rho = 0.4$ | 25 | 522 | 65.45 ± 8.44 | 1169 | 263 |
| $\rho = 0.6$ | 50 | 364 | 65.89 ± 8.26 | 1081 | 329 |
| **Temperature = 1.3** | | | | | |
| Baseline ($\rho = 0$) | 6 | 87 | 55.14 ± 18.39 | 188 | 0 |
| $\rho = 0.2$ | 8 | 54 | 59.95 ± 16.33 | 150 | 36 |
| $\rho = 0.4$ | 7 | 40 | 62.29 ± 12.85 | 139 | 39 |
| $\rho = 0.6$ | 12 | 31 | 65.15 ± 8.42 | 132 | 49 |

Table 9: 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. ± Std.) | Gen. Budget | Prune Budget |
|---|---|---|---|---|---|
| **Insurance Dataset** (Abdelghany, 2021) | | | | | |
| Baseline ($\rho = 0$) | 8 | 760 | 62.66 ± 10.72 | 1261 | 0 |
| $\rho = 0.2$ | 11 | 604 | 64.13 ± 10.66 | 1108 | 200 |
| $\rho = 0.4$ | 20 | 477 | 64.68 ± 9.54 | 1041 | 234 |
| $\rho = 0.6$ | 48 | 328 | 67.21 ± 8.26 | 960 | 296 |
| $\rho = 0.8$ | 140 | 140 | 69.76 ± 6.98 | 840 | 420 |
| **Diabetes Dataset** (Khare, 2020) | | | | | |
| Baseline ($\rho = 0$) | 8 | 725 | 65.74 ± 10.10 | 1208 | 0 |
| $\rho = 0.2$ | 10 | 520 | 66.02 ± 11.03 | 975 | 175 |
| $\rho = 0.4$ | 21 | 444 | 67.74 ± 9.06 | 977 | 221 |
| $\rho = 0.6$ | 44 | 312 | 68.66 ± 7.34 | 917 | 281 |
| $\rho = 0.8$ | 134 | 134 | 67.56 ± 7.39 | 804 | 402 |
| **Education Dataset** (Fatima, 2023) | | | | | |
| Baseline ($\rho = 0$) | 8 | 800 | 62.37 ± 10.97 | 1320 | 0 |
| $\rho = 0.2$ | 11 | 584 | 63.12 ± 10.18 | 1074 | 194 |
| $\rho = 0.4$ | 22 | 501 | 64.97 ± 8.87 | 1082 | 245 |
| $\rho = 0.6$ | 48 | 346 | 67.08 ± 7.47 | 1007 | 309 |
| $\rho = 0.8$ | 147 | 147 | 66.98 ± 8.27 | 882 | 441 |
| **Shopping Dataset** (Siddiqa, 2022) | | | | | |
| Baseline ($\rho = 0$) | 8 | 805 | 62.37 ± 11.51 | 1348 | 0 |
| $\rho = 0.2$ | 14 | 624 | 63.47 ± 10.24 | 1172 | 212 |
| $\rho = 0.4$ | 22 | 501 | 63.57 ± 10.62 | 1110 | 249 |
| $\rho = 0.6$ | 50 | 342 | 64.25 ± 10.09 | 1023 | 313 |
| $\rho = 0.8$ | 150 | 150 | 65.55 ± 10.36 | 900 | 450 |
| **Happiness Dataset** (Buttar, 2023) | | | | | |
| Baseline ($\rho = 0$) | 8 | 715 | 65.15 ± 8.98 | 1251 | 0 |
| $\rho = 0.2$ | 10 | 552 | 65.98 ± 8.46 | 1042 | 186 |
| $\rho = 0.4$ | 20 | 450 | 65.23 ± 9.35 | 1012 | 226 |
| $\rho = 0.6$ | 46 | 320 | 67.02 ± 8.16 | 971 | 295 |
| $\rho = 0.8$ | 139 | 139 | 70.06 ± 8.91 | 834 | 417 |

Table 10: Robustness of Selective TTS in generalizing across five diverse datasets, all of which exhibit consistent improvements under scaling.

| Setting | Runs | Total Final Reports | Score (Avg. ± Std.) | Gen. Budget | Prune Budget | Avg. Insight Tokens |
|---|---|---|---|---|---|---|
| **Original Prompt, ∼85 tokens** | | | | | | |
| Baseline ($\rho = 0$) | 8 | 805 | $61.20 \pm 13.02$ | 1403 | 0 | $84.50 \pm 19.62$ |
| $\rho = 0.2$ | 12 | 644 | $63.08 \pm 10.78$ | 1222 | 218 | $85.74 \pm 19.09$ |
| $\rho = 0.4$ | 25 | 522 | $65.45 \pm 8.44$ | 1169 | 263 | $88.50 \pm 20.50$ |
| $\rho = 0.6$ | 50 | 364 | $65.89 \pm 8.26$ | 1081 | 329 | $91.50 \pm 20.99$ |
| $\rho = 0.8$ | 156 | 156 | $64.91 \pm 8.41$ | 936 | 468 | $93.42 \pm 20.57$ |
| **∼130 tokens** | | | | | | |
| Baseline ($\rho = 0$) | 8 | 795 | $65.80 \pm 13.34$ | 1369 | 0 | $129.72 \pm 23.08$ |
| $\rho = 0.2$ | 12 | 608 | $69.74 \pm 8.54$ | 1168 | 208 | $134.56 \pm 23.61$ |
| $\rho = 0.4$ | 21 | 501 | $69.57 \pm 8.77$ | 1116 | 249 | $134.81 \pm 24.43$ |
| $\rho = 0.6$ | 48 | 362 | $71.48 \pm 6.91$ | 1061 | 323 | $139.72 \pm 25.46$ |
| $\rho = 0.8$ | 152 | 152 | $71.79 \pm 6.70$ | 912 | 456 | $148.73 \pm 25.50$ |
| **∼180 tokens** | | | | | | |
| Baseline ($\rho = 0$) | 8 | 740 | $71.15 \pm 10.51$ | 1281 | 0 | $177.74 \pm 35.48$ |
| $\rho = 0.2$ | 12 | 552 | $72.23 \pm 10.06$ | 1062 | 190 | $174.76 \pm 35.72$ |
| $\rho = 0.4$ | 20 | 489 | $73.98 \pm 7.21$ | 1071 | 240 | $178.06 \pm 36.00$ |
| $\rho = 0.6$ | 45 | 330 | $74.54 \pm 6.49$ | 980 | 298 | $191.45 \pm 40.73$ |
| $\rho = 0.8$ | 142 | 142 | $75.49 \pm 6.66$ | 852 | 426 | $197.73 \pm 39.15$ |
| **∼220 tokens** | | | | | | |
| Baseline ($\rho = 0$) | 10 | 790 | $74.06 \pm 10.44$ | 1376 | 0 | $222.87 \pm 51.68$ |
| $\rho = 0.2$ | 15 | 604 | $74.97 \pm 8.78$ | 1166 | 210 | $220.49 \pm 54.22$ |
| $\rho = 0.4$ | 27 | 502 | $76.71 \pm 6.43$ | 1124 | 256 | $235.08 \pm 62.04$ |
| $\rho = 0.6$ | 53 | 352 | $77.47 \pm 5.49$ | 1051 | 323 | $242.85 \pm 58.36$ |
| $\rho = 0.8$ | 153 | 153 | $78.92 \pm 5.52$ | 918 | 459 | $246.76 \pm 55.07$ |

Table 11: Robustness to output length variation under different insight-generation constraints.

Insight-only Pruning performs slightly better than Random Pruning but remains significantly weaker than Selective TTS, demonstrating that pruning applied only at the final stage is insufficient to realize the benefits of structured test-time selection.

### E.5 Detailed Results for Robustness Analyses

This section provides the full numerical results corresponding to the robustness studies reported in § 4.4. Across three dimensions—budget formulation, including model backbone, and decoding configuration, we observe that Selective TTS produces highly consistent scaling trends, supporting the reliability and generality of the method.

**Token-Level vs. LLM-Call Budget.** Figure 7 visualizes the score and variance trends when the total compute is controlled using output tokens rather than LLM calls. Table 7 reports the detailed statistics, including the number of runs, final reports, and token-level budgets. Across all pruning ratios $\rho$, the performance curves under token budgeting closely match those obtained under LLM-call budgeting: mean scores rise from $\rho = 0$ to $\rho = 0.6$, before slightly declining at $\rho = 0.8$, and variance follows the same decreasing-then-increasing pattern. The number of runs and total generated tokens also exhibit similar trends. These observations confirm that LLM calls provide a faithful and stable proxy for token-level compute, while additionally offering easier control and higher reproducibility in practice.

**Cross-Model Variation.** To examine whether scaling behavior depends on the underlying model family, we evaluate Selective TTS under four generator–judger backbone combinations: Q–G, G–Q, Q–Q, and G–G. The grouped-bar comparison in Figure 8 shows that all four backbones exhibit consistent improvement as $\rho$ increases, though the absolute score levels differ across models. Table 8 provides the full numerical breakdown for each pruning ratio. The similar upward trends across model families indicate that Selective TTS does not rely on a specific generator or judger, and that no strong family-specific bias is driving the performance gains.

**Sensitivity to Decoding Parameters.** We further test robustness to decoding diversity by varying the generation temperature while keeping top-$p$ fixed. Figure 9 summarizes the score improvements across temperatures, and Table 9 lists the detailed results 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. While extremely high temperatures introduce instability and reduce the number of usable runs (as reflected in Appendix tables), the overall scaling pattern remains stable: Selective TTS continues to allocate compute effectively even as output diversity shifts.

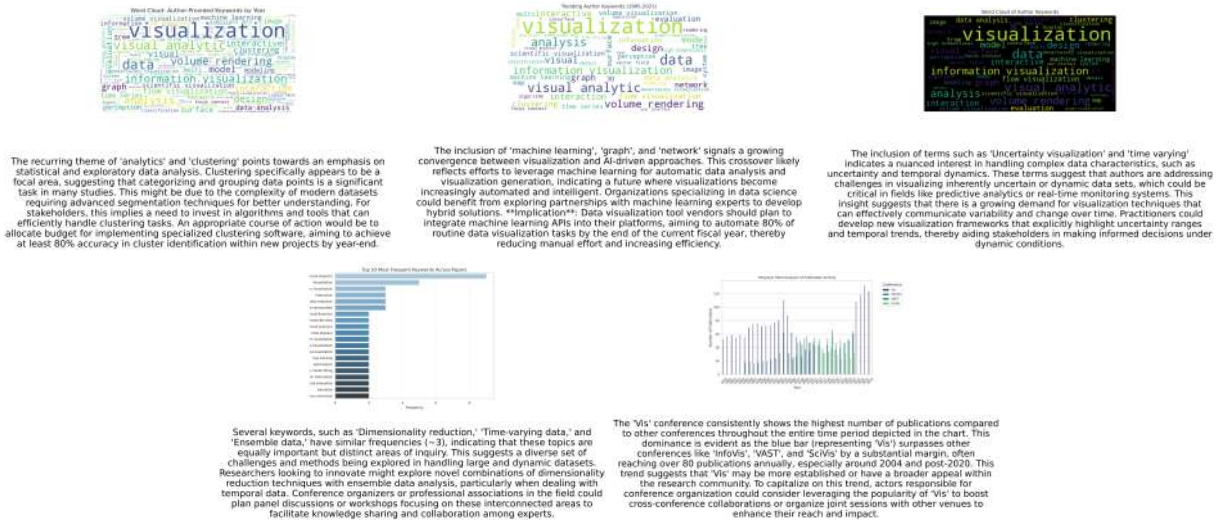



Figure 18: Low-scoring reports across all pruning ratios. The corresponding scores are: $\rho$, with scores of $\rho = 0$: 11.3, $\rho = 0.2$: 20, $\rho = 0.4$: 11.3, $\rho = 0.6$: 32.9, $\rho = 0.8$: 47.5.

These results demonstrate that Selective TTS is robust to decoding variability and does not depend on a narrow set of generation parameters.

**Generalization Across Datasets.** We further assess robustness by evaluating Selective TTS on six additional tabular datasets (Insurance, Diabetes, Education, Shopping, Happiness, and Student). Figure 10 visualizes the score trends across pruning ratios, and Table 10 reports the full numeric results.

Across all datasets, Selective TTS consistently improves performance as $\rho$ increases from 0 to 0.6, mirroring the behavior observed on the VIS dataset. The aggressive setting ($\rho = 0.8$) produces dataset-dependent outcomes: several datasets continue to improve, while others show weaker gains, suggesting that heavy pruning may remove useful branches when data are noisier or structurally complex. Overall, these results indicate that Selective TTS generalizes reliably across diverse data domains, with moderate pruning ratios offering the most stable improvements.

**Sensitivity to Insight Length.** Figure 11 reveals a clear association between insight length and final evaluation score under the baseline setting ($\rho = 0$), motivating an investigation into whether Selective TTS remains effective when the length of generated insights varies.

Specifically, 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 categories. Table 11 reports the detailed numerical results for each regime, while Figure 12 summarizes the corresponding performance trends. Across all insight-length settings, Selective TTS consistently improves performance as the pruning ratio $\rho$ increases, and the relative scaling trends remain stable. 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 insight stage fails to achieve comparable or stable improvements, indicating that length-based selection at the final stage is insufficient. Second, within each length regime, the variance in average insight length across $\rho \in \{0, 0.4\}$ is relatively small (e.g., ∼4 tokens for the ∼85-token setting, ∼5 tokens for ∼130 tokens, ∼1 token for ∼180 tokens, and ∼13 tokens for ∼220 tokens), yet meaningful performance improvements are still observed.

| **Reports** | Kendall's $\tau$ (↑) | Spearman's $\rho$ (↑) | Kendall's W |
|---|---|---|---|
| Low | 0.70±0.10 | 0.82±0.08 | 0.7875 |
| Medium | 0.30±0.41 | 0.38±0.45 | 0.2750 |
| High | -0.15±0.17 | -0.17±0.19 | 0.1250 |

Table 12: Alignment metrics between the harsh judger 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.

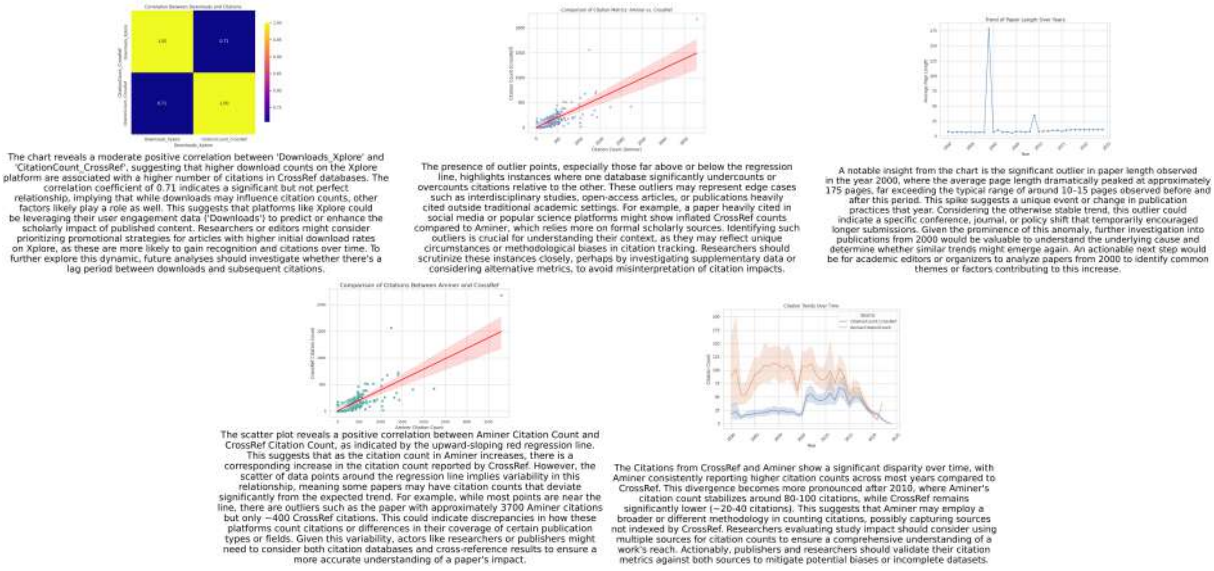



Figure 19: High-scoring reports across all pruning ratios. The corresponding scores are: $\rho = 0$: 85.4, $\rho = 0.2$: 85, $\rho = 0.4$: 84.6, $\rho = 0.6$: 84.2, $\rho = 0.8$: 84.2.

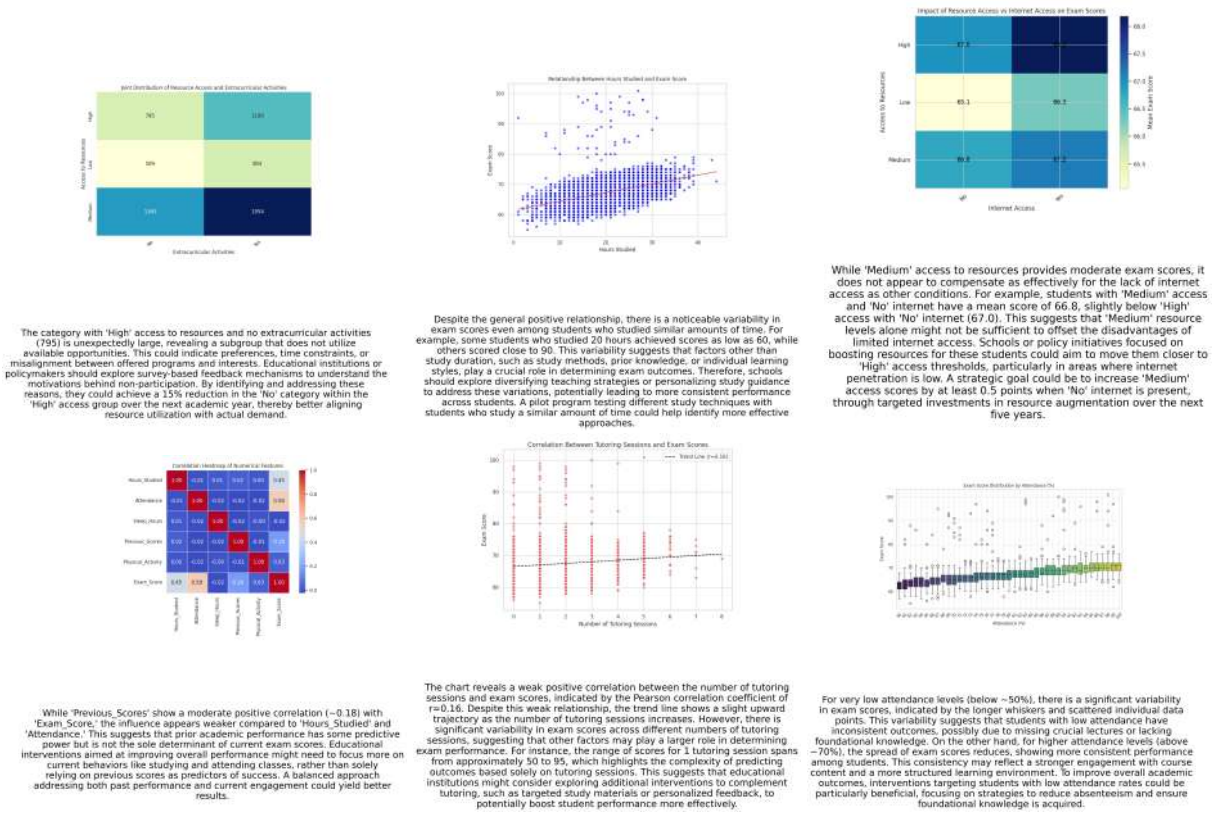



Figure 20: Example reports for Education Performance Dataset

## 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% percentiles 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 per-

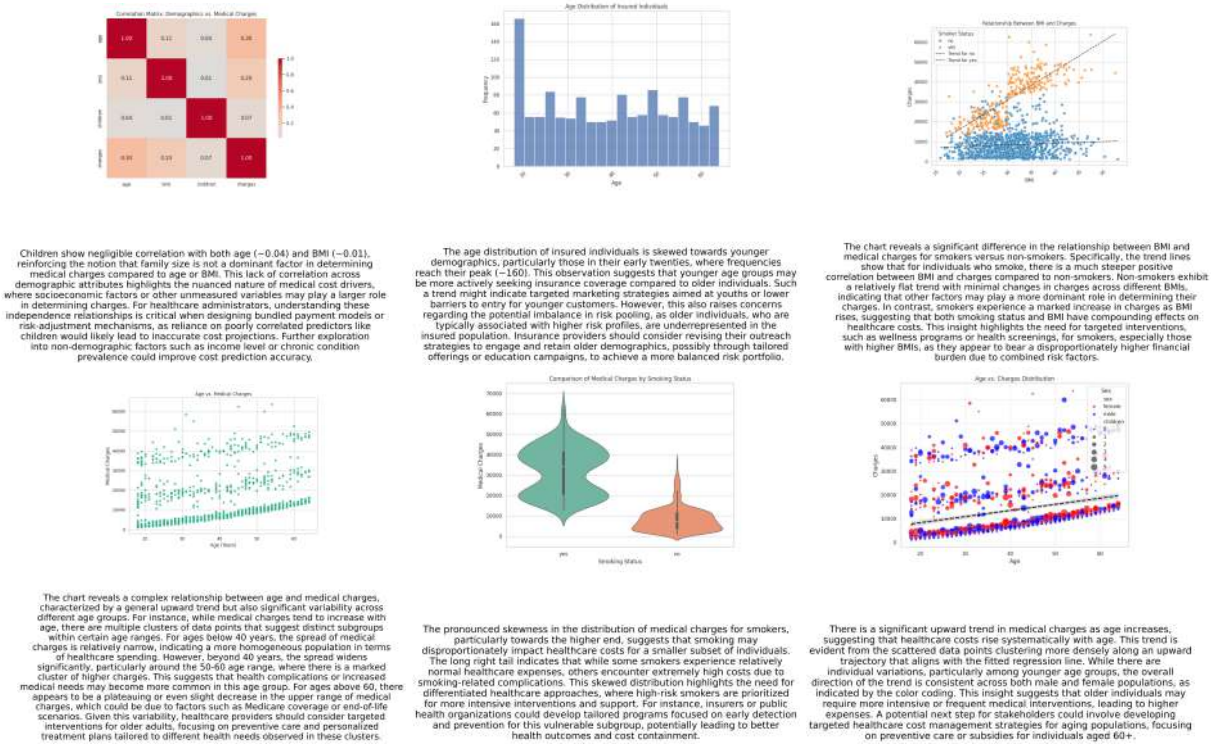



Figure 21: Example reports for Medical Insurance Dataset

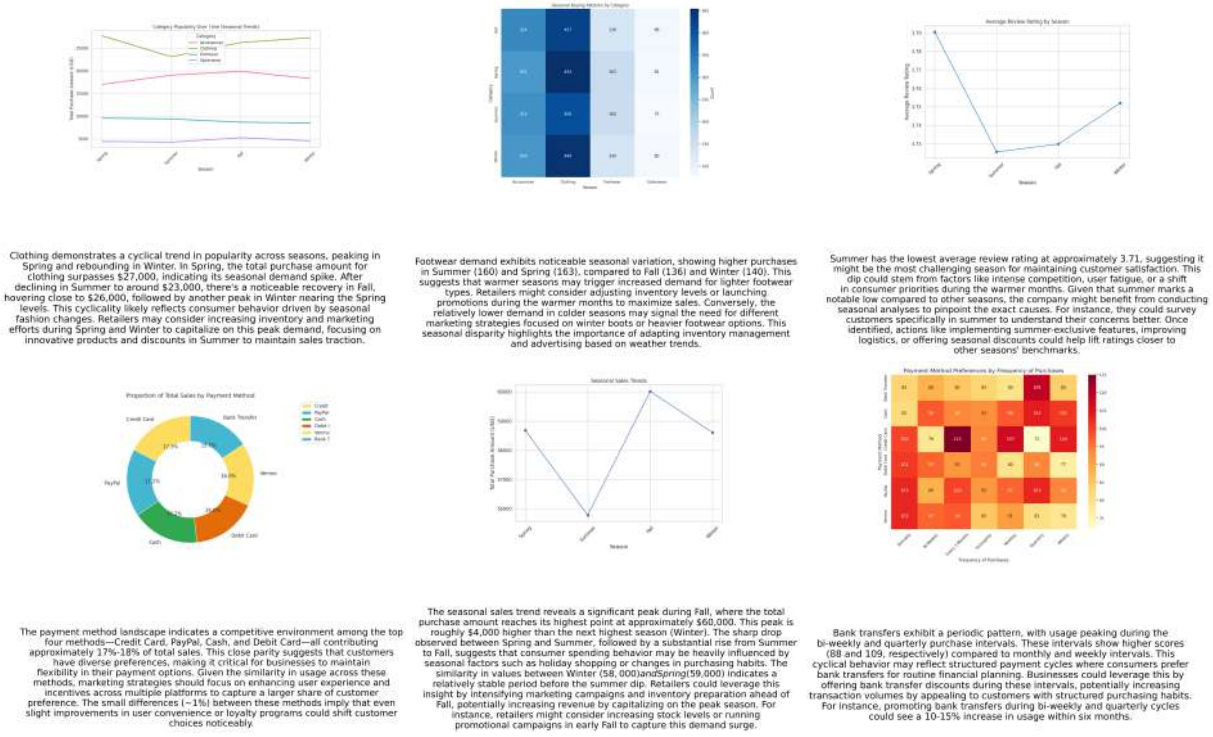
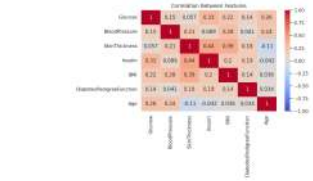
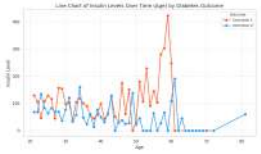


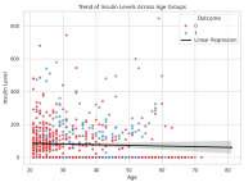

Figure 22: Example reports from Shopping Behavior Dataset



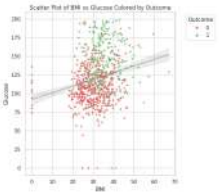
The DiabetesPedigreeFunction exhibits a notably low correlation with most features, including Glucose (0.14), BloodPressure (0.041), SkinThickness (0.18), Insulin (0.19), and BMI (0.14). This finding implies that family history or genetic susceptibility to diabetes, as represented by the DiabetesPedigreeFunction, 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.



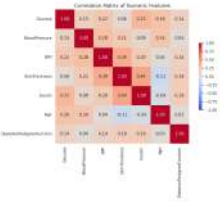
The sharp increase in insulin levels observed in Outcome 1 around age 60 suggests a significant metabolic shift that could indicate the onset of severe diabetic complications. This peak, exceeding 400 units, stands out as an anomaly compared to the consistent trend in Outcome 0, where insulin levels 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.



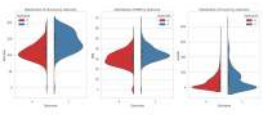
The chart illustrates a substantial overlap in insulin levels between 'Outcome 0' and 'Outcome 1' across various age groups. While there is a discernible separation in younger age groups, the lines for both outcomes become increasingly indistinguishable as age progresses. This overlap becomes particularly 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 higher. 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.

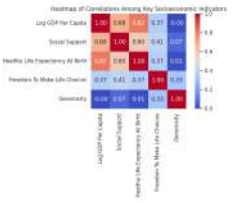


Glucose shows moderate correlations with several features: BMI (0.22), BloodPressure (0.15), and Insulin (0.33). This pattern indicates that glucose levels are influenced by a combination of body composition and metabolic activity. The presence of these correlations may highlight the complex interplay 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.

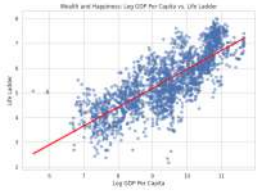


The relationship between glucose and insulin levels across the outcomes reveals an indirect linkage tied to metabolic health. While glucose levels differ sharply between outcomes 0 and 1, insulin levels also exhibit a distinct pattern indicative of different physiological states. For outcome 0, the low 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.

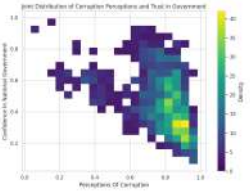
Figure 23: Example reports from Diabetes Dataset



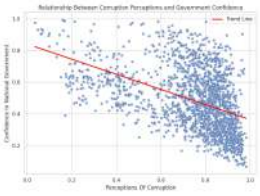
The heatmap reveals a strong positive correlation between 'Log GDP Per Capita' and both 'Social Support' and 'Healthy Life Expectancy At Birth,' with correlation coefficients of 0.68 and 0.82, respectively. This suggests that higher income levels within a country are closely linked to enhanced social support systems and better health outcomes. The relatively moderate positive correlation with 'Freedom To Make Life Choices' (0.37) indicates that economic prosperity may also contribute to personal freedoms, though to a lesser extent compared to social support and life expectancy. These findings imply that policy interventions focused on improving GDP per capita might simultaneously boost various social and health indicators. However, the lack of significant correlation with 'Generosity' (-0.00) could suggest that wealth alone does not necessarily translate to increased generosity in societal behavior. A possible intervention strategy could involve reallocating funds towards social programs that emphasize community-building activities to encourage more generous behaviors alongside economic growth initiatives.



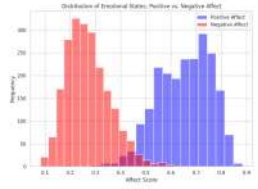
There appears to be a diminishing marginal return effect of GDP on happiness as we move to higher GDP per capita levels. While the red regression line shows a positive correlation, the slope begins to flatten as the Log GDP Per Capita increases toward the right end of the x-axis. This implies that while economic growth significantly enhances happiness initially, its impact plateaus at higher GDP levels. This observation aligns with the concept of the 'Easterlin Paradox,' which posits that beyond a certain level of affluence, further economic gains do not substantially improve subjective well-being. Policymakers targeting happiness maximization should focus on ensuring equitable distribution of wealth rather than solely pursuing economic growth once a country reaches a certain prosperity threshold. For example, redistributive policies and investments in social services may yield better results in enhancing happiness compared to further GDP-driven strategies.



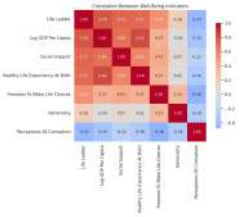
There is a negative correlation between perceptions of corruption and confidence in the national government, as indicated by the concentration of high-density areas (yellow-green regions) towards the bottom-right and top-left quadrants of the heatmap. Specifically, as the perceptions of corruption increase (approaching 1.0), confidence in the government decreases, and vice versa. This suggests that public perception of corruption significantly influences trust levels in governance. The density distribution demonstrates that the majority of data points lie along this trend, highlighting the importance of addressing corruption to restore public trust. Policymakers should focus on implementing transparent anti-corruption measures to shift the population towards higher confidence levels, using interventions targeted at regions with high corruption perceptions (e.g., a policy aimed at reducing corruption levels by 0.2 points within two years).



The data demonstrates a non-linear drop in confidence as perceptions of corruption rise. While the decline is gradual at lower corruption levels (below 0.3), the rate of decline accelerates significantly once perceptions exceed 0.5. This sharp decrease suggests that public tolerance for corruption has a threshold beyond which confidence rapidly diminishes. Such non-linearity may indicate that citizens are willing to overlook minor instances of corruption but become highly sensitive to more pervasive issues. Policymakers should therefore focus intensively on preventing corruption from crossing this threshold to avoid sudden trust collapses. Developing early detection systems and rapid response mechanisms for emerging corruption could be key strategies to maintain confidence.



The Negative Affect distribution exhibits a steep decline in frequency beyond a score of 0.3. This sharp drop-off is evident as the red bars reduce dramatically towards the right side of the histogram. This pattern suggests that as affect scores increase, the likelihood of experiencing negative emotions decreases substantially, indicating a potential saturation or 'ceiling effect' for negative affect at higher scores. This could imply that individuals who report higher affect scores are less likely to report significant negative experiences, pointing to a possible threshold where positivity dominates. Public health campaigns focused on improving overall well-being might aim to guide individuals towards achieving affect scores above 0.3, as doing so seems to correlate with a marked reduction in negative emotional experiences. Strategic initiatives could measure success by tracking the shift in mean affect scores moving above this critical threshold.



The correlation matrix shows that 'Social Support' has moderately strong positive correlations with 'Life Ladder' (0.72) and 'Log GDP Per Capita' (0.68), suggesting that social cohesion and support systems contribute to both subjective well-being and economic prosperity. However, the relatively weak correlation with 'Generosity' (0.07) raises questions about whether social support independently drives charitable behavior or if it requires additional factors. For example, cultivating a sense of community through public policy could enhance social support, which in turn might indirectly encourage generosity by fostering empathy. A potential strategy could involve local governments investing in community-building initiatives, aiming to boost social support scores by 30% within five years, thereby exploring indirect pathways to influence generosity.

Figure 24: Example reports from World Happiness Report Dataset

ception. 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).** Figure 18 compares low-scoring reports across all pruning ratios. The differences are more pronounced than in the high-quality group (Figure 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 rankings showed larger dispersion among these low-quality samples (Figure 13), suggesting that raters could easily distinguish their relative strengths.

**Cross-pruning comparisons (high-quality group).** In contrast, Figure 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, while $\rho$=0.4 analyzes paper-length anomalies). Annotators rated these reports comparably strong, as reflected in the small variance of mean rankings in Figure 13.

# G Qualitative Report Examples on Additional Datasets

To complement the quantitative robustness analyses, we provide qualitative examples of reports generated by Selective TTS 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.

Figure 20 shows representative reports generated on the Education Performance dataset. Figure 21 presents example reports from the Medical Insurance dataset. Figure 22 displays reports generated on the Shopping Behavior dataset. Figure 23 provides examples from the Diabetes dataset. Finally, Figure 24 shows reports generated on the World Happiness Report dataset.

# H Pipeline Design

This section provides an extended description of our multi-agent pipeline architecture and its core design components. 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 describe the prompt design and specifications for all major agents and judgers involved in each stage, including both task-level generation and stage-local evaluation modules (§H.2).

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

Figure 25 presents the full workflow of our proposed multi-agent data science pipeline, expanding upon the conceptual stages illustrated earlier in Figure 2. While Figure 2 provides a high-level overview of the agents and their interconnections, Figure 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 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 judgers 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 Figure 25, the core pipeline comprises six generation components: the *Data Profiling*, *Visualizatio 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
```

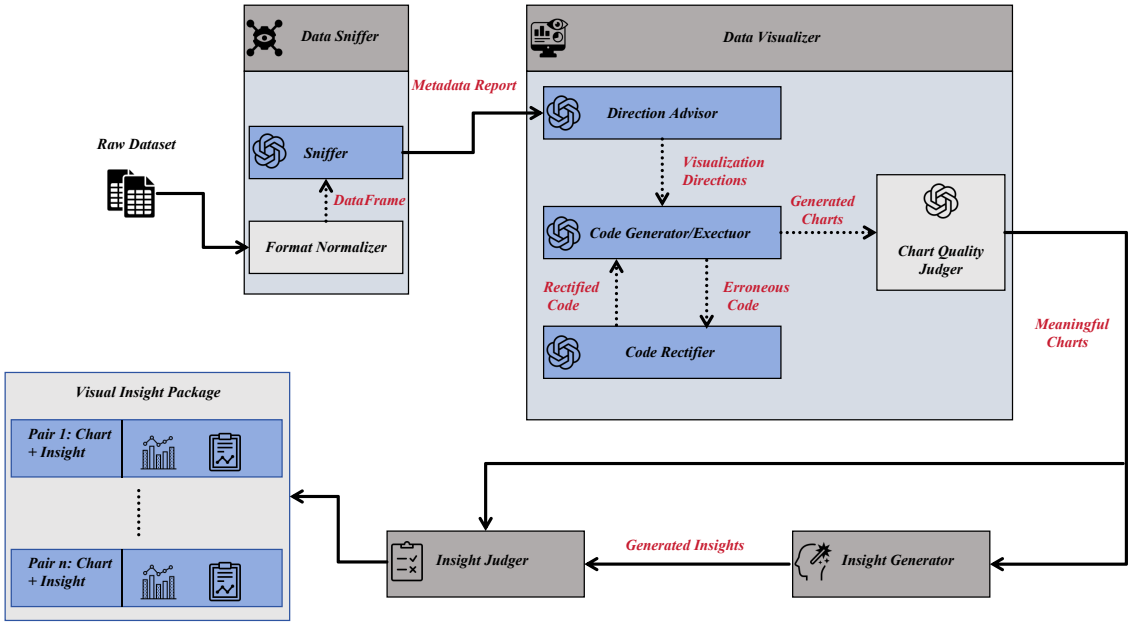



Figure 25: Detailed Pipeline of Insight Generation.

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 ∈ {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.

## 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 Judgers for Final Evaluation.** We employ three *multi-level judgers*: **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 judgers produce structured outputs in JSON format with detailed subscores, 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** 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 & 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. Trivally 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.**