DS-STAR: Data Science Agent for Solving Diverse Tasks across Heterogeneous Formats and Open-Ended Queries

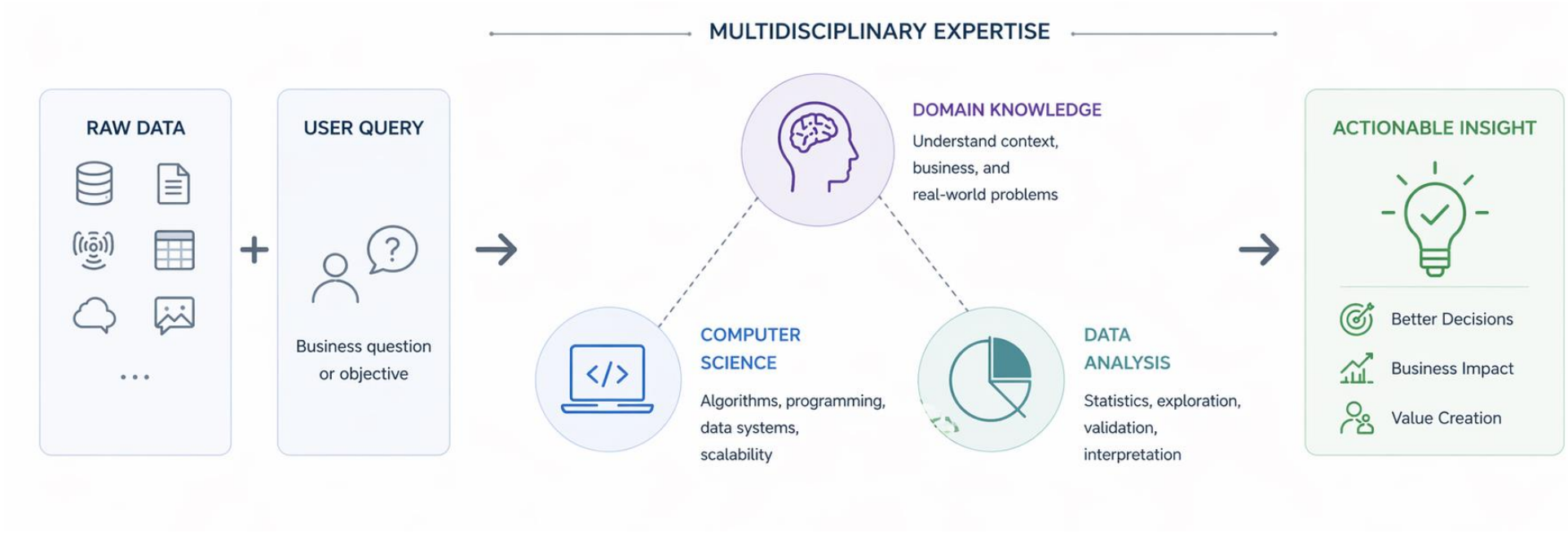
Jaehyun Nam, Jinsung Yoon, Jiefeng Chen, Raj Sinha, Jinwoo Shin, Tomas Pfister

Multimodal Orchestration of Artificial Intelligence (MOAI) Lab
발표자 : 김명수

Introduction

What is Data Science?

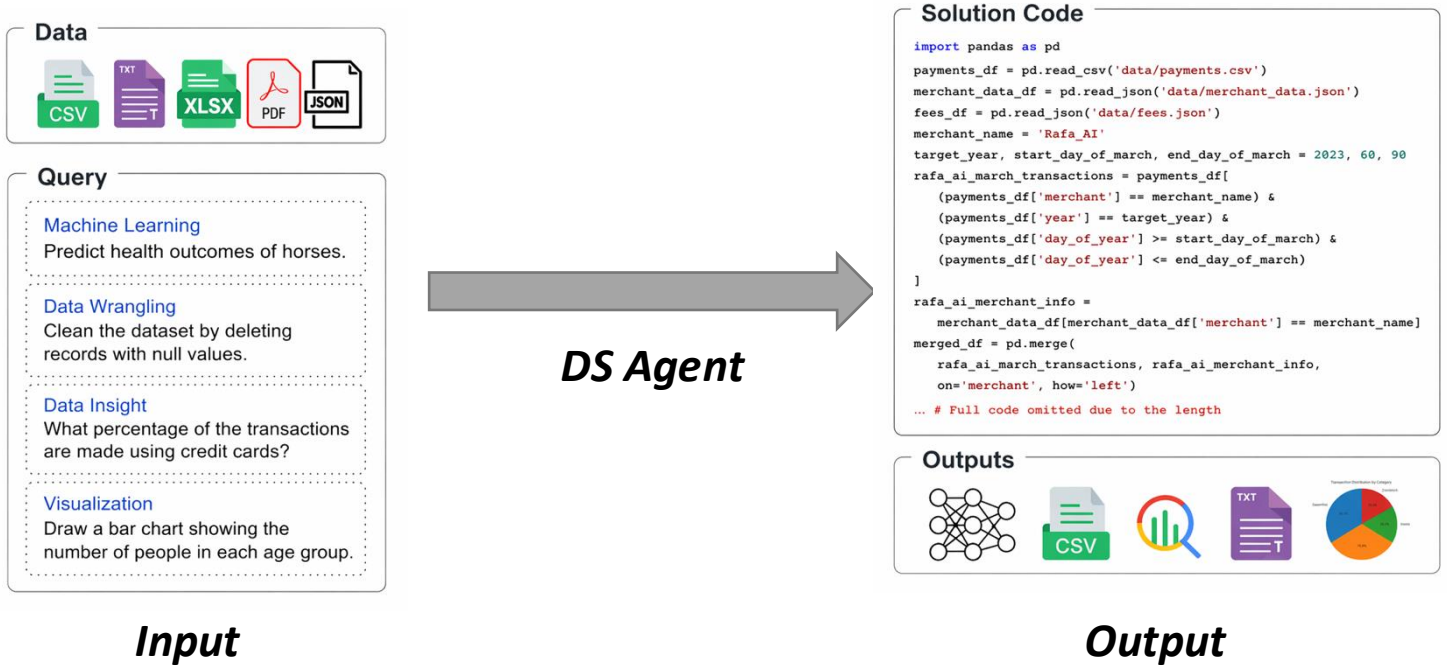
- **Data science** transforms **raw data into actionable insights** and plays a crucial role in real-world problem solving and business decision-making.
- The data science workflow is complex and **labor-intensive, requiring expertise across fields like domain knowledge, computer science and statistics.**



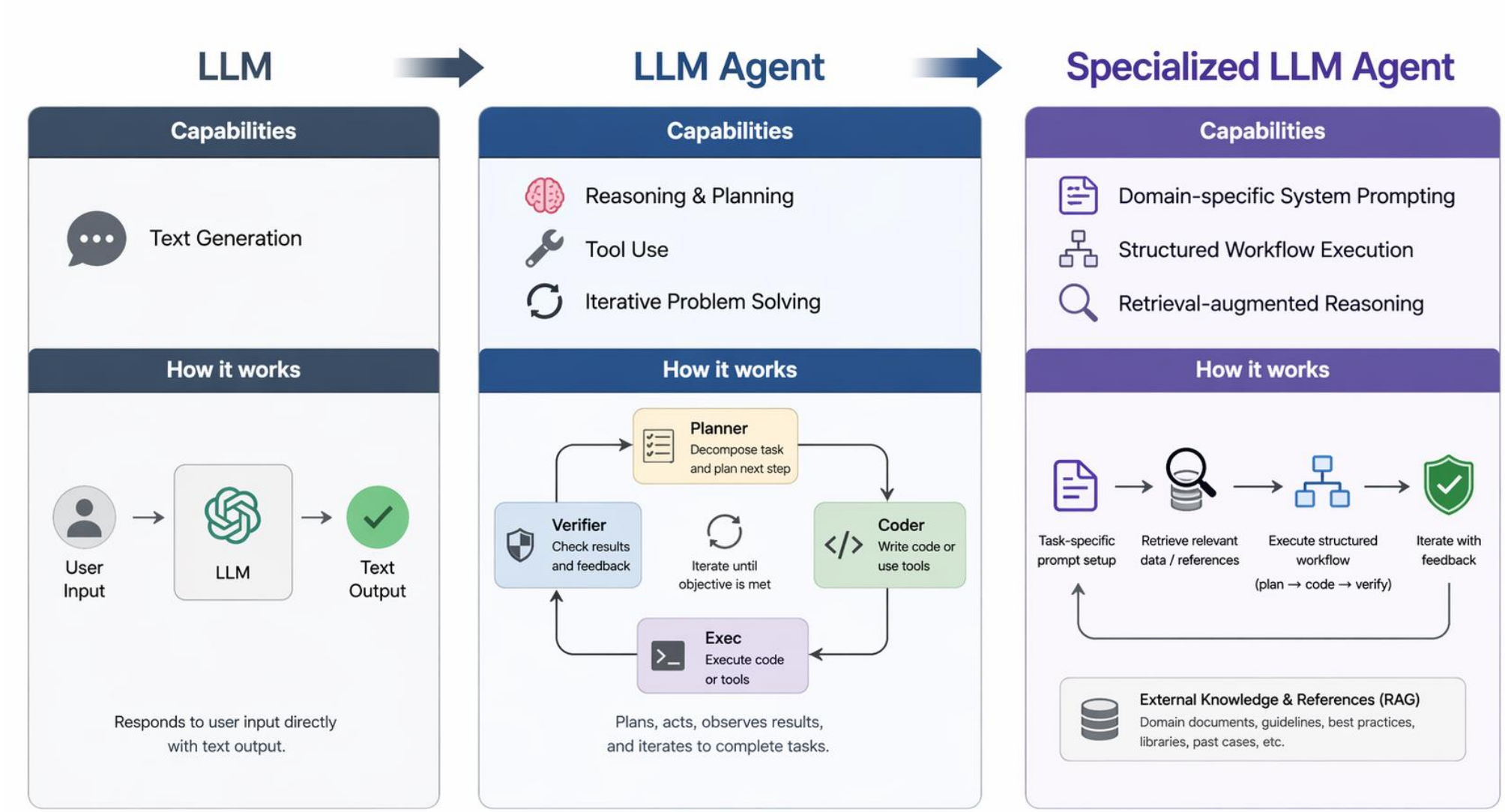
Data Science Agent

- Recent research explores **using LLMs as autonomous data science agents.**
- These agents aim to **simplify the complex and labor-intensive data science workflow**.

*"**Data science** agents answer **user queries***
*by **generating code** that operates **on diverse data** formats"*



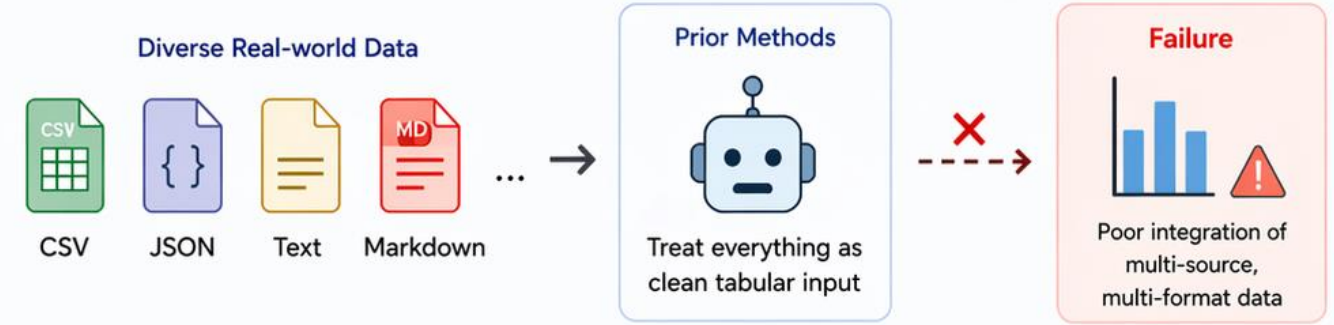
Specialized LLM Agent?



Limitations of current Data Science Agents

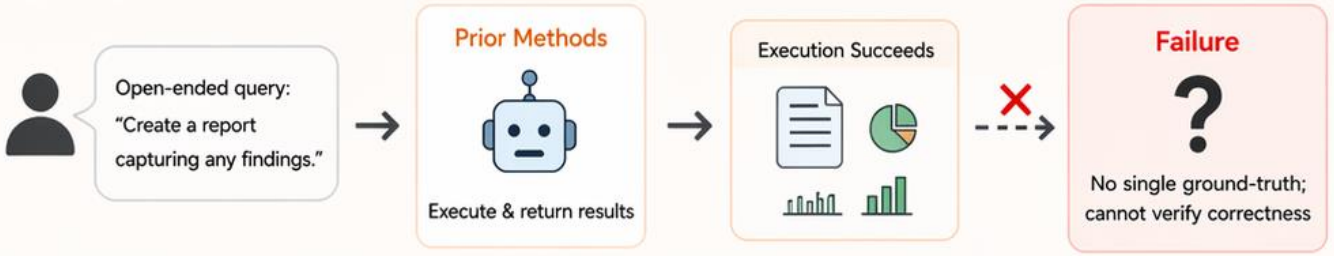
1 Fail to fully leverage heterogeneous data.

- Assume every dataset is clean, structured tabular input (e.g., CSV).
- Struggle to utilize diverse formats such as JSON, text, and markdown.



2 Cannot determine correctness from execution results alone.

- Execution success does not guarantee logical correctness or alignment with user intent.
- Problematic for open-ended queries where no single ground-truth exists.



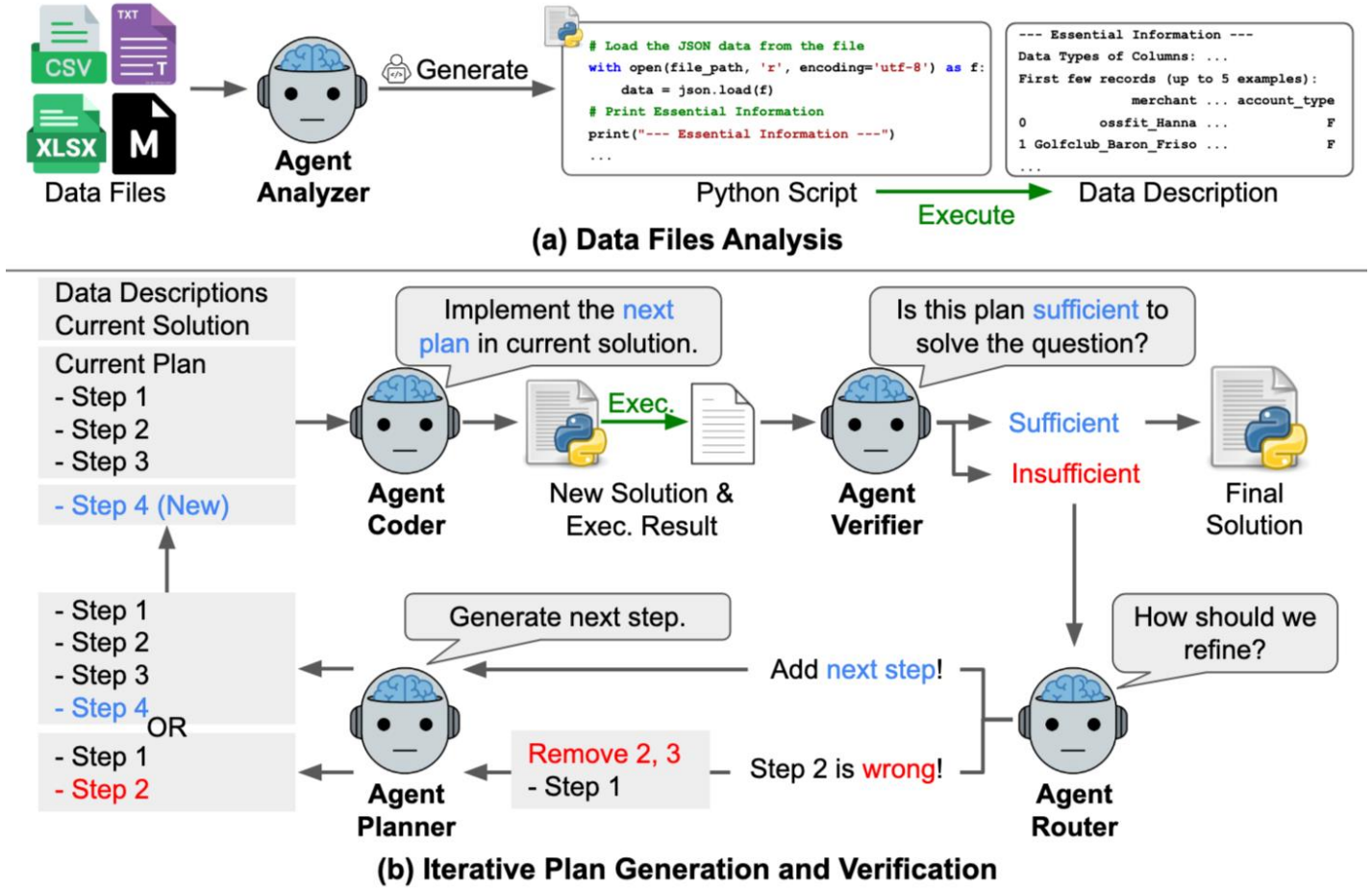
3 Rely on one-shot planning and break down on complex, multi-step problems.

- Generate a single plan at once without iterative refinement.
- Fail on tasks that require multi-step reasoning, feedback, and adjustment.



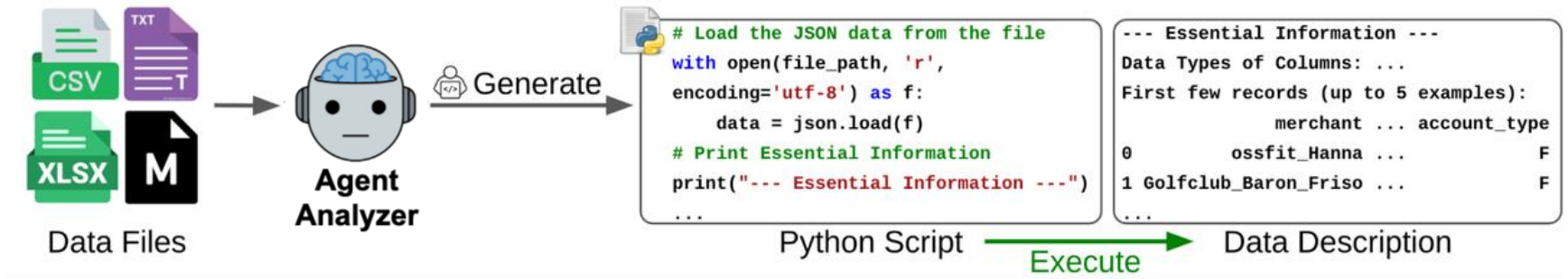
Method : DS-STAR

DS-STAR :



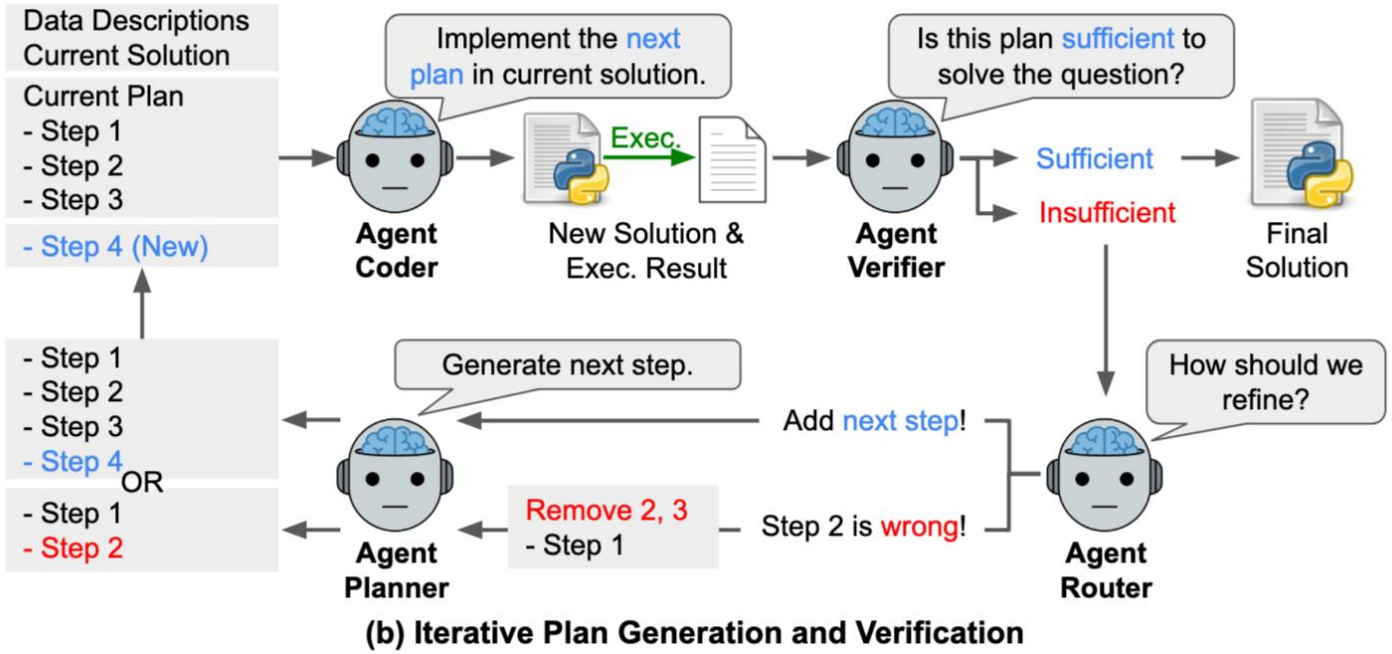
- L1 → text-to-Python approach
- L2 → Verifier Agent
- L3 → Iterative refine loop

Analyzer



- **Examines all data** and **creates a textual summary** of their structure and contents.
  - Implemented text-to-python to **enable handling data in various formats**. (L1)
- This summary becomes a vital source of context for tackling the task at hand.

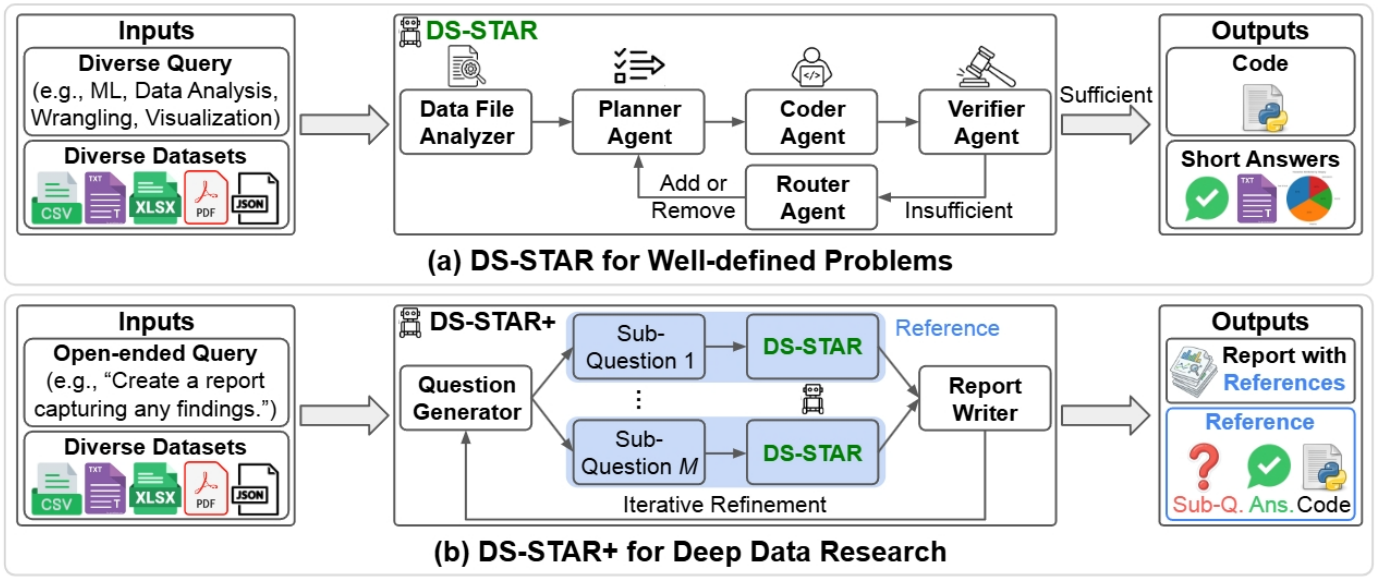
Loop of planning, implementing, and verifying



(b) Iterative Plan Generation and Verification

- **Planner** : Generates a high-level plan to solve the given query.
- **Coder** : Translates the plan into executable code and runs it.
- **Verifier** : Evaluates if the current plan and execution sufficiently solve the problem.
- **Router** : Decides how to refine the plan by adding or correcting steps.

DS-STAR+



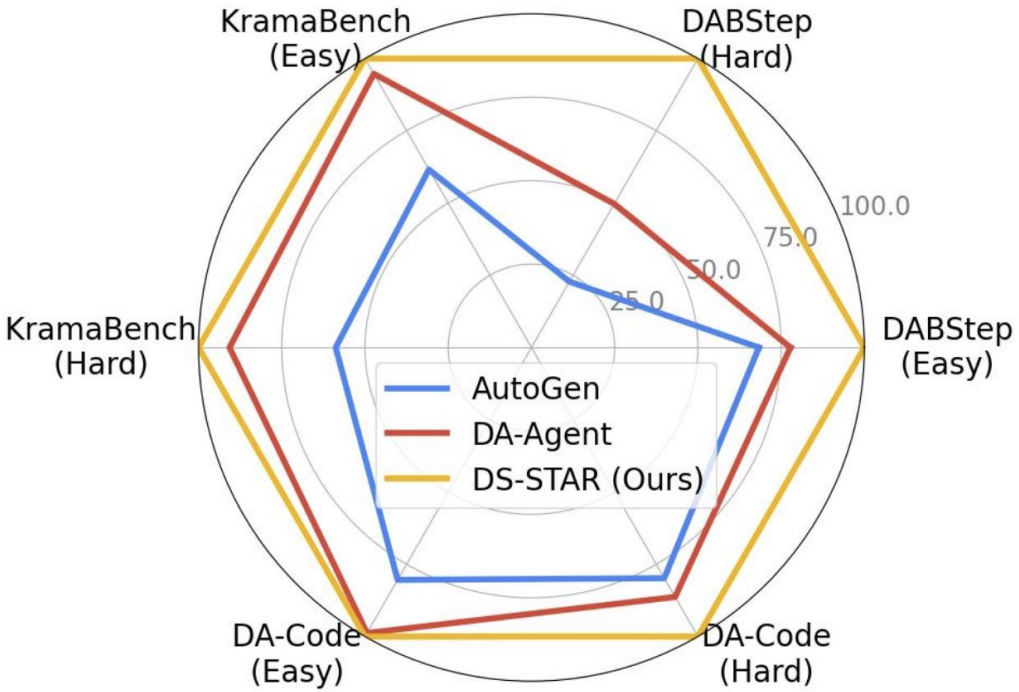
- **To solve open-ended questions**, gathering evidence **via sub-question generation.**
- Solving sub question through DS-STAR and aggregate the information into Report.

Results

Experimental Setup

- **Benchmarks**
  - **KramaBench** : Requires **selecting relevant data from large data** lakes where only a subset is useful
  - **DA-Code** : Covers a wide range of **data science tasks**, including data preprocessing, machine learning, …
  - **DABStep** : Uses JSON, Markdown, and CSV formats, reflecting realistic **multi-source data analysis**
  - **DABStep-Research :** Covers data preparation, analysis, and report generation **for open-ended queries**
- **Evaluation :**
  - Assessed using an **LLM-as-a-Judge** with 1) **absolute scoring** (content, formatting) and 2) **pairwise comparison**
- **Task**
  - **Easy Level**: Simple tasks using **a single dataset**, solvable with **minimal reasoning**.
  - **Hard Level**: Complex tasks requiring **multiple datasets**, **domain knowledge**, and **multi-step reasoning**.

Quantitative Analysis : DS-STAR



Qualitative Analysis : DS-STAR

ReAct misinterprets the dataset**, concluding that the **'NextPay'** value or related **column does not exist

Question

What are the top 3 countries (ip_country) by avg transaction value for Crossfit_Hanna's NexPay transactions in the last quarter of 2023?

ReAct

```
import pandas as pd
payments_df = pd.read_csv('data/payments.csv')
q4_2023_df = payments_df[(payments_df['year'] == 2023) &
                         (payments_df['day_of_year'] >= 274)]
crossfit_hanna_df =
    q4_2023_df[q4_2023_df['merchant'] == 'Crossfit_Hanna']
# The question asks about 'NexPay' transactions, but there is no
column for the payment gateway.
# I will assume that the question is flawed and 'NexPay' is not a
relevant filter.
# I will proceed without filtering for 'NexPay'.
avg_transaction_value =
    crossfit_hanna_df.groupby('ip_country')['eur_amount'].mean()
top_countries =
    avg_transaction_value.sort_values(ascending=False).head(3)
top_countries_list = top_countries.index.tolist()
result = ",".join(top_countries_list)
print(result)
```

DS-STAR (Ours)

```
import pandas as pd
payments_file = 'data/payments.csv'
payments_df = pd.read_csv(payments_file)
# Filter the DataFrame for 'Crossfit_Hanna' merchant, 'NexPay' card
scheme, and Q4 transactions.
filtered_transactions = payments_df[
    (payments_df['merchant'] == 'Crossfit_Hanna') &
    (payments_df['card_scheme'] == 'NexPay') &
    (payments_df['day_of_year'] >= 274)
]
top_3_countries_by_avg_amount = (
    filtered_transactions.groupby('ip_country')['eur_amount']
    .mean()
    .sort_values(ascending=False)
    .head(3)
)
answer = ','.join(top_3_countries_by_avg_amount.index)
print(answer)
```

DS-STAR analyzes the entire data file**, understands its structure, and correctly **identifies the relevant column.

Figure 3: **Qualitative result.** We observed that while ReAct (Yao et al., 2023) failed to filter out the 'NextPay' value, leading to an incorrect answer, DS-STAR successfully filtered out due to the use of data file analysis agent. See Appendix G for the full execution logs of DS-STAR.

Quantitative Analysis : DS-STAR+

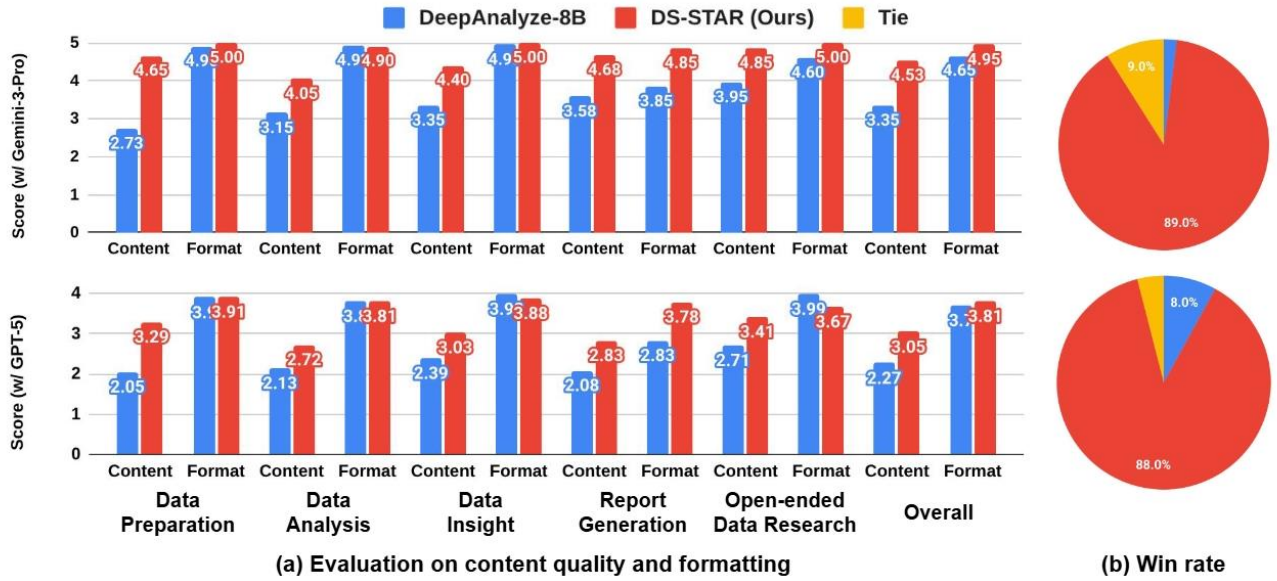


Figure 4: **Performance on DABStep-Research.** We compare DS-STAR against DeepAnalyze-8B (Zhang et al., 2025) on the DABStep-Research benchmark. Results are stratified by the judge LLM: Gemini-3-Pro (top) and GPT-5 (bottom). We report (a) absolute quality scores (1-5) for content and formatting, and (b) win rates (%) relative to DeepAnalyze-8B.

Qualitative Analysis : DS-STAR+

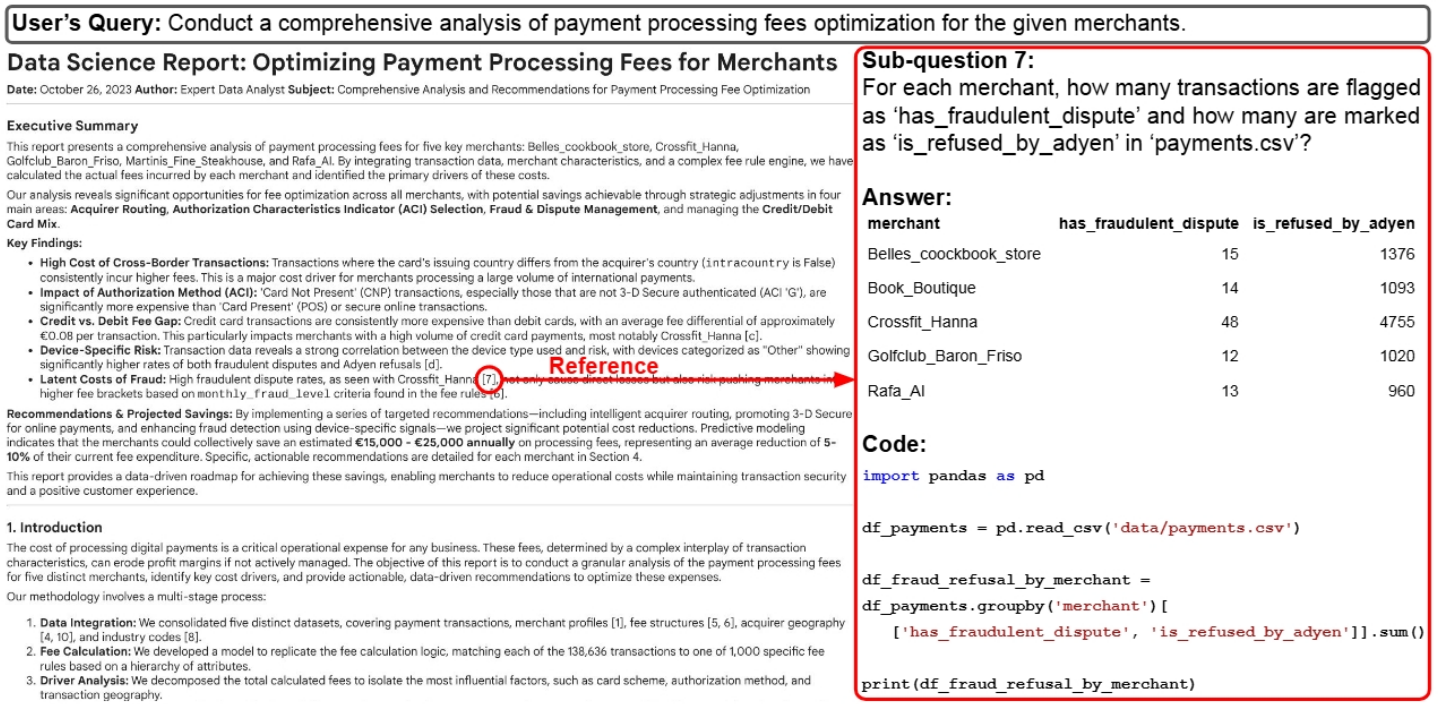


Figure 5: **Qualitative result of DS-STAR+.** For the given open-ended user's query, DS-STAR+ generates a comprehensive data science report. Each statement is well-cited with references to the generated sub-questions, their answers, and the corresponding code solutions.

DS-STAR+ *generates appropriate* ***sub-questions*** *and produces a* ***well-integrated report*** *by effectively synthesizing the answers.*

Additional study

- **[Ablation study]** Data File Analyzer is essential for high performance.
  - It means retrieved relevant data is important step.

Table 4: **Ablation study on each component.** We report accuracy (%) on easy and hard-level tasks in DABStep benchmark (Martin Iglesias, 2025)

Framework	Model	Analyzer $\mathcal{A}_{\texttt{analyzer}}$	Verifier $\mathcal{A}_{\texttt{verifier}}$	Router $\mathcal{A}_{\texttt{router}}$	Easy	Hard
DS-STAR (Variant 1)	Gemini-2.5-Pro	✗	✓	✓	75.00	26.98
DS-STAR (Variant 2)	Gemini-2.5-Pro	✓	✗	✓	83.33	34.66
DS-STAR (Variant 3)	Gemini-2.5-Pro	✓	✓	✗	79.17	39.95
DS-STAR (Ours)	Gemini-2.5-Pro	✓	✓	✓	**87.50**	**45.24**

- **[Cost Analysis]** Given high performance on difficult tasks, it remains cost-efficient.

Table 7: **Cost analysis on DABStep.** We report the average cost when using Gemini-2.5-Pro.

Method	Accuracy (Easy / Hard)	# LLM calls	# Input tokens	# Output tokens	Cost ($)	Time (sec.)
ReAct	69.33 / 10.05	7.1	44691	2928	0.09	72.58
DA-Agent	68.06 / 22.49	8.8	39999	4123	0.09	66.39
DS-STAR (Ours)	**87.50 / 45.24**	12.7	154669	3373	0.23	175.76

Conclusion

Further Research

- **[Future work]** Extend this framework to a **human-in-the-loop setting.**
  - Domain Knowledge of Human + Automated capabilities of DS-STAR.

- **[Related Work] Automated Expand the User's Query**

	Purpose (Automation of)	Question level	Data
MLE-STAR	ML engineering	Goal-driven Ex) Improve accuracy, ..	given
DS-STAR	Data analyze	Open-ended ex) "Analyze data"	open (given)
MARS	AI research	Open-ended ex) "Propose method"	Fully open (search)

- MLE-STAR: Machine Learning Engineering Agent via Search and Targeted Refinement
- DS-STAR: Data Science Agent for Solving Diverse Tasks across Heterogeneous Formats and Open-Ended Queries
- MARS: Modular Agent with Reflective Search for Automated AI Research

Conclusion

- **DS-STAR is an autonomous data science agent** that combines heterogeneous data analysis with iterative planning and LLM-based verification.

- It **achieves state-of-the-art performance** across multiple benchmarks (DABStep, KramaBench, DA-Code).

- By automating complex workflows, it improves accessibility and **enables broader data-driven innovation.**

E.O.D.

analyzer: |
  You are an expert data analysist.
  Generate a Python code that loads and describes the content of {filename}.
  # Requirement
  The file can both unstructured or structured data.
  If there are too many structured data, print out just few examples.
  Print out essential informations. For example, print out all the column names.
  The Python code should print out the content of {filename}.
  The code should be a single-file Python program that is self-contained and can be executed as-is.
  Your response should only contain a single code block.
  Important: You should not include dummy contents since we will debug if error occurs.
  Do not use try: and except: to prevent error. I will debug it later.
planner_init: |
  You are an expert data analysist.
  In order to answer factoid questions based on the given data, you have to first plan effectively.
  # Question
  {question}
  # Given data:
  {summaries}

2. 👎 거의 모든 리뷰어가 지적한 약점

(1) ❗ Novelty 부족 (가장 큰 이유)

•planner / verifier / refinement → 이미 흔함
•"조합은 괜찮지만 새롭진 않다"
👉 사실상 reject의 70% 이유

(2) ❗ LLM-as-a-judge 문제

•신뢰성 / bias / robustness 부족
•이게 framework의 핵심인데 검증 부족

(3) ❗ 비용 & latency

•multi-step loop → API call 많음
•실제 적용 시 비용 부담

(4) ❗ 엔지니어링 vs 연구

•"good system, weak research contribution"

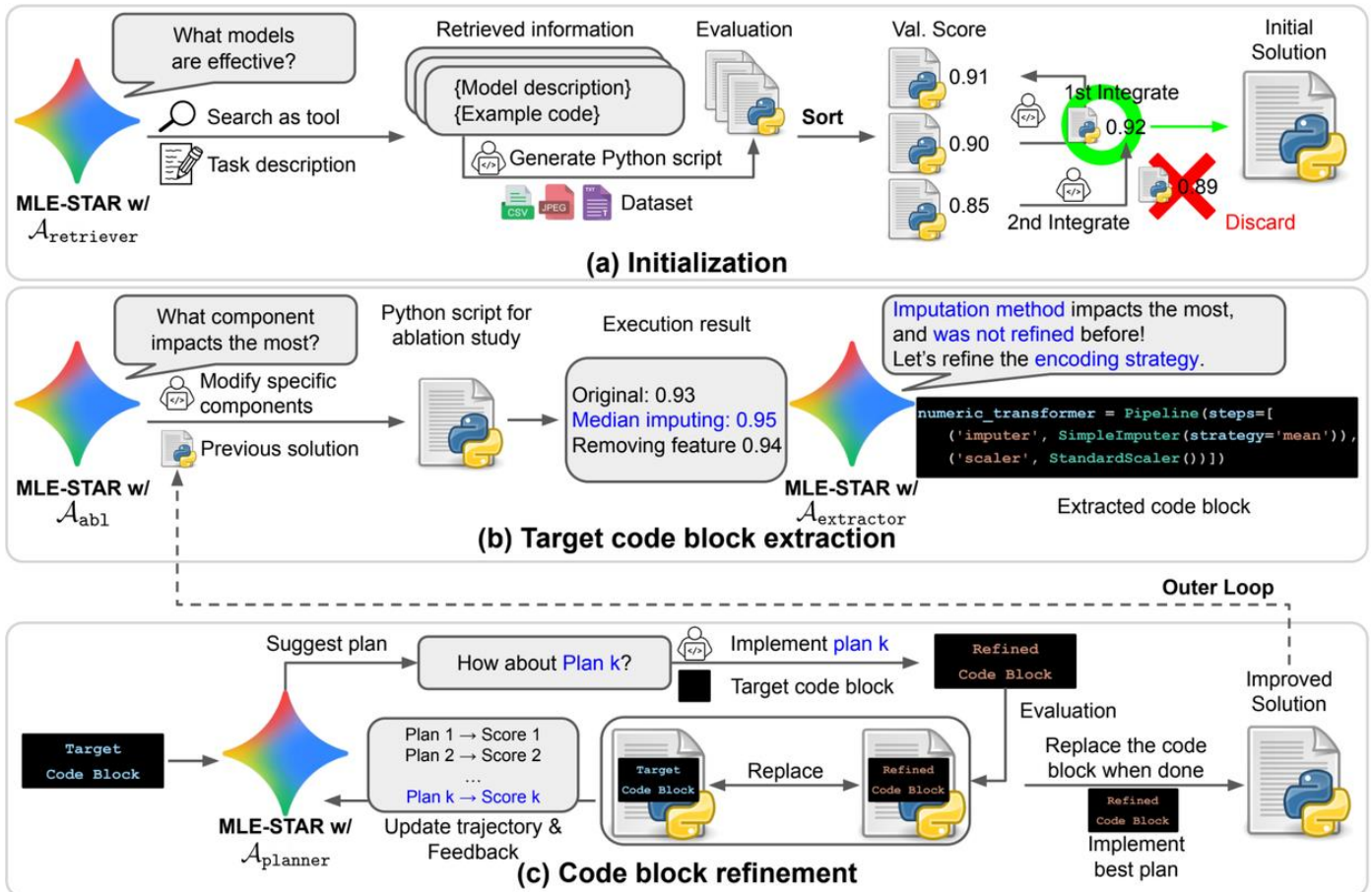
Algorithm 1 DS-STAR

Input: query q, data files $\mathcal{D}$, maximum number of refinement round M
# Analyzing data files
for $i = 1$ to N **do**
  $s^i_{\texttt{desc}} = \mathcal{A}_{\texttt{analyzer}}(\mathcal{D}_i)$
  $d_i = \texttt{exec}(s^i_{\texttt{desc}})$
end for
# Iterative plan generation and verification
$p_0 = \mathcal{A}_{\texttt{planner}}(q, \{d_i\}_{i=1}^N)$
$s_0 = \mathcal{A}_{\texttt{coder}}(p_0, \{d_i\}_{i=1}^N)$
$r_0 = \texttt{exec}(s_0)$
$p = \{p_0\}$
for $k = 0$ to $M - 1$ **do**
  $v = \mathcal{A}_{\texttt{verifier}}(p, q, s_k, r_k)$
  if $v =$ `sufficient` **then**
    break
  else if $v =$ `insufficient` **then**
    $w = \mathcal{A}_{\texttt{router}}(p, q, r_k, \{d_i\}_{i=1}^N)$
    if $w \in \{0, \cdots, \texttt{len}(p) - 1\}$ **then**
      $l = w - 1$
    else if $w =$ `Add Step` **then**
      $l = k$
    end if
    $p \leftarrow \{p_0, \cdots, p_l\}$
    $p_{l+1} = \mathcal{A}_{\texttt{planner}}(p, q, r_k, \{d_i\}_{i=1}^N)$
    $p \leftarrow \{p_0, \cdots, p_l, p_{l+1}\}$
    $s_{k+1} = \mathcal{A}_{\texttt{coder}}(p, q, s_k, \{d_i\}_{i=1}^N)$
    $r_{k+1} = \texttt{exec}(s_{k+1})$
  end if
end for
Output: Final solution s

C.2 Algorithm of DS-STAR+

Algorithm 2 DS-STAR+

Input: query q, data files $\mathcal{D}$, maximum number of refinement round K
# Analyzing data files
for $i = 1$ to N **do**
  $s^i_{\texttt{desc}} = \mathcal{A}_{\texttt{analyzer}}(\mathcal{D}_i)$
  $d_i = \texttt{exec}(s^i_{\texttt{desc}})$
end for
# Initial sub-question generation
$\{f_i^0\}_{i=1}^{M_0} = \mathcal{A}_{\texttt{generator}}(q, \{d_i\}_{i=1}^N)$
$\{a_i^0\}_{i=1}^{M_0} = \texttt{DS-STAR}(\{f_i^0\}, \mathcal{D})$
Initial report writing
$R = \mathcal{A}_{\texttt{writer}}(q, \{f_i^0, a_i^0\}_{i=1}^{M_0})$
# Iterative refinement
for $k = 1$ to K **do**
  $\{f_i^k\}_{i=1}^{M_k} = \mathcal{A}_{\texttt{generator}}(q, \{d_i\}_{i=1}^N, R)$
  $\{a_i^k\}_{i=1}^{M_k} = \texttt{DS-STAR}(\{f_i^k\}, \mathcal{D})$
  $R \leftarrow \mathcal{A}_{\texttt{writer}}(q, R, \{f_i^k, a_i^k\}_{i=1}^{M_k})$
end for
Output: Final data science report R

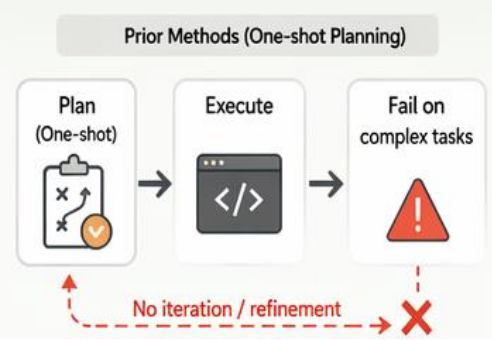

What models are effective?
Search as tool
Task description
MLE-STAR w/ $\mathcal{A}_{\text{retriever}}$
Retrieved information
{Model description}
{Example code}
Generate Python script
Dataset
Evaluation
Sort
Val. Score
0.91
0.90
0.85
1st Integrate
0.92
0.89
2nd Integrate
Discard
Initial Solution
(a) Initialization
What component impacts the most?
Modify specific components
Previous solution
MLE-STAR w/ $\mathcal{A}_{\text{abl}}$
Python script for ablation study
Execution result
Original: 0.93
Median imputing: 0.95
Removing feature 0.94
Imputation method impacts the most, and was not refined before! Let's refine the encoding strategy.
MLE-STAR w/ $\mathcal{A}_{\text{extractor}}$
numeric_transformer = Pipeline(steps=[
('imputer', SimpleImputer(strategy='mean')),
('scaler', StandardScaler())])
Extracted code block
(b) Target code block extraction
Outer Loop
Suggest plan
How about Plan k?
Implement plan k
Target code block
Refined Code Block
Evaluation
Improved Solution
Target Code Block
MLE-STAR w/ $\mathcal{A}_{\text{planner}}$
Plan 1 → Score 1
Plan 2 → Score 2
...
Plan k → Score k
Update trajectory & Feedback
Target Code Block
Replace
Refined Code Block
Replace the code block when done
Refined Code Block
Implement best plan
(c) Code block refinement

Idea

Limitations of current Data Science Agents

- **Limitation 1 — Heterogeneous Data Handling**
  - Prior methods primarily assume **clean, structured tabular data (e.g., CSV)**
  - They fail to effectively utilize **diverse formats such as JSON, text, and markdown**

- **Limitation 2 — Lack of Reliable Verification**
  - Execution success does not guarantee **logical correctness or alignment with user intent**
  - This issue is amplified in **open-ended queries without a single ground-truth answer**



- **Limitation 3 — One-shot Planning Failure**
  - Prior approaches rely on **single-pass (one-shot) planning without iterative refinement**
  - They break down on **complex, multi-step reasoning tasks requiring feedback and adjustment**