University of Central Florida

CAP6545 - Machine Learning for Biomedical Data

DS-Agent
Automated Data Science

Empowering Large Language Models with Case-Based Reasoning

Authors: Siyuan Guo, Cheng Deng, Ying Wen, Hechang Chen, Yi Chang, Jun Wang

Presented by: Hieu Chu

Challenges in Automating Data Science

- Data science requires a mix of skills—statistics, coding, and domain knowledge—making it time-consuming and demanding.
- Current LLMs agents (AutoGPT, LangChain, ResearchAgent) struggle with automating complicated machine learning (ML) tasks, especially those needing multiple steps or detailed reasoning.
- Many LLM agents generate unrealistic and ineffective experiment plans, showing the limitations in problem-solving.
- These issues slow down progress, reduce efficiency, and limit how widely data-driven insights can be used in different industries.

Unlocking the Power of AI in Data Science

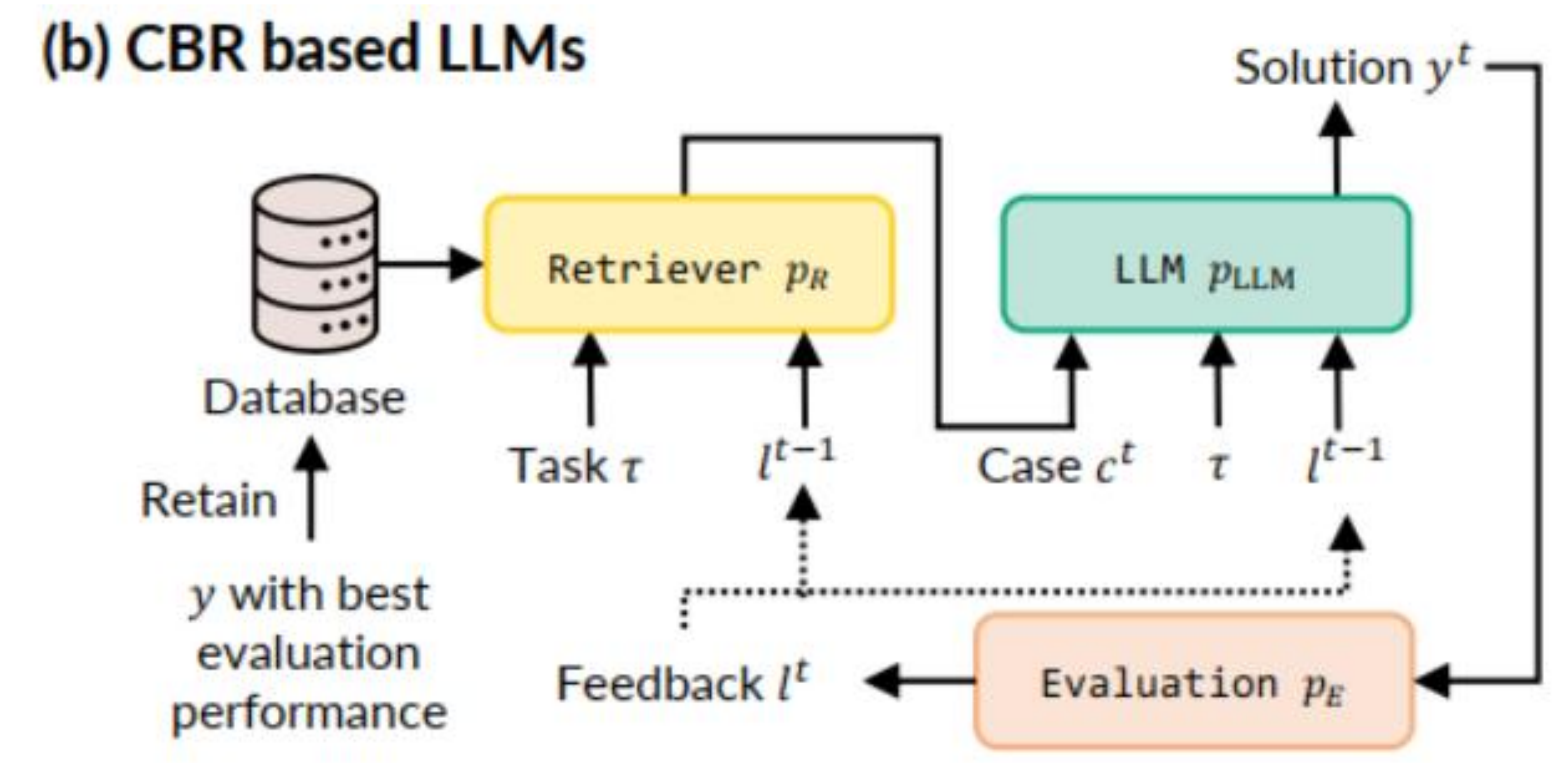
Expanding Access to Data Insights

- Enables people with limited technical skills to analyze data effectively.
- Provides a reliable AI assistant for data science tasks.
- Makes machine learning more accessible across different industries.
- Reduces the need for extensive technical knowledge to build ML models.

A Smarter Approach for LLMs

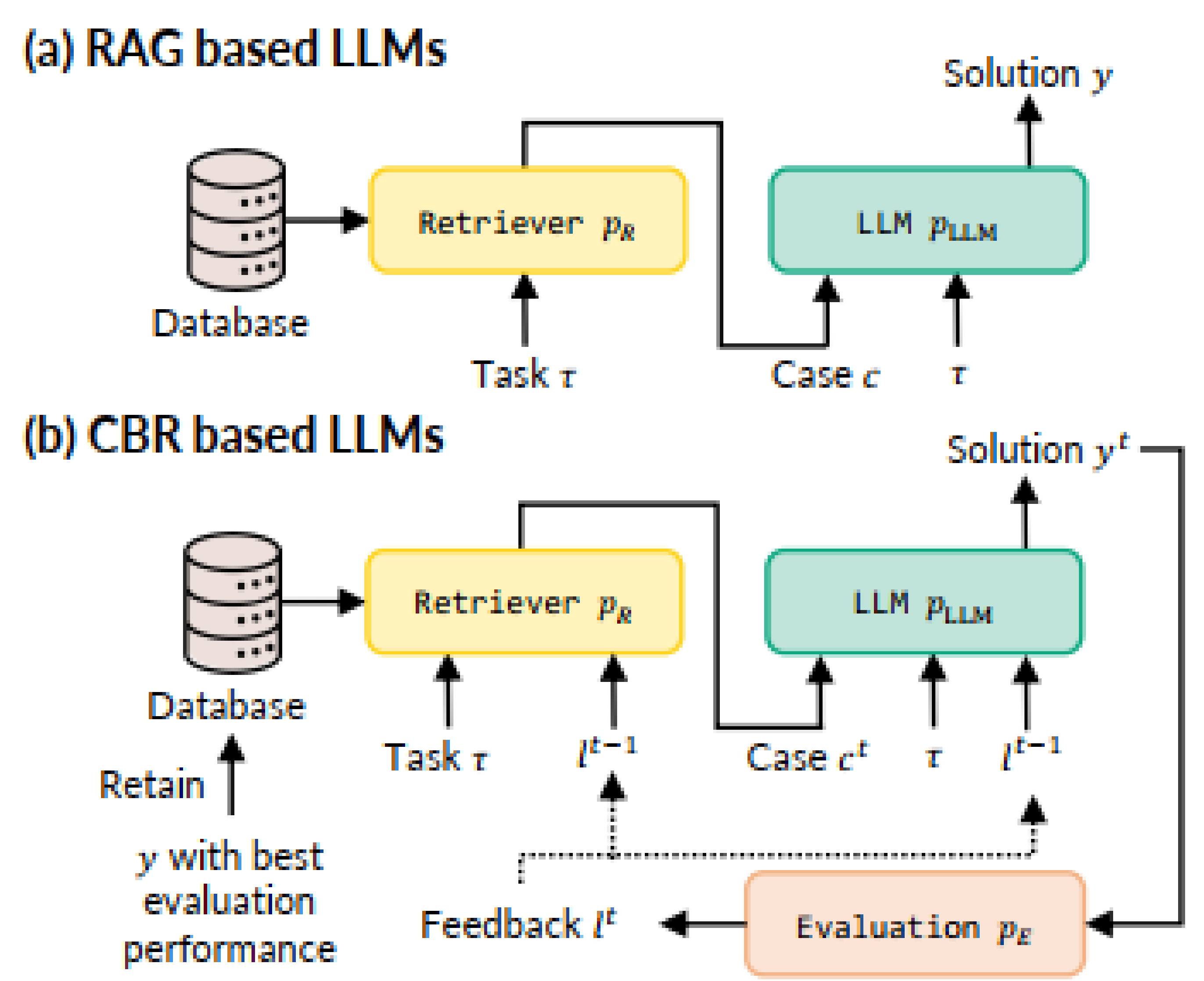
Boosting AI with Human Expertise

- Kaggle offers a large repository of expert knowledge through reports and code.
- DS-Agent combines LLMs with Case-Based Reasoning (CBR), allowing it to learn from past human successes.
- Uses feedback from task execution to improve performance step by step.
- Offers a flexible way to learn and adapt without heavy fine-tuning, while maintaining high accuracy.



Case-Based Reasoning (CBR) in DS-Agent

- A classical AI paradigm that solves new problems by retrieving, adapting, evaluating, and iteratively refining solutions from similar past cases.
- DS-Agent uses CBR to leverage expert insights from Kaggle, refining solutions through feedback for better performance.
- Unlike RAG, CBR features a feedback loop for iterative refinement and reuses successful cases for future tasks.



Where the Data Comes From and How It's Used

- Expert solutions, report and top-ranking code from Kaggle competitions and Research paper. Detailed reports describe methods, while code snippets offer the implementation.
- **30 data science task** includes text, time-series data, and structured tables with two fundamental task type of **regression** and **classification**.
- Divide dataset to **Development** and **Deployment** Stage, follow the DS-Agent workflow
- Use nature language to describe each task that can handle by LLMs models and create baseline framework of python script.

Task Description

```
You are solving this machine learning tasks of regression:
The dataset presented here (the Airline reviews) comprises customer feedback for British Airways. Here, we
     provide the textual reviews. Your task is to predict the corresponding rating in the range of {1, ...,
     10} given the reviews in the test set. The evaluation metric is root mean squared error (RMSE).
We provide an overall pipeline in train.py. Now fill in the provided train.py script to train a language
     model to get a good performance.
```

The provided Python script (train.py)

```python
import pandas as pd
from sklearn.metrics import mean_squared_error
import numpy as np
import random
import torch
from sklearn.model_selection import train_test_split
from submission import submit_predictions_for_test_set

SEED = 42
random.seed(SEED)
torch.manual_seed(SEED)
np.random.seed(SEED)
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")

def compute_metrics_for_regression(y_test, y_test_pred):
    rmse = mean_squared_error(y_test, y_test_pred, squared=False)
    return rmse

def train_model(X_train, y_train, X_valid, y_valid):
    # TODO. define and train the model
    # should return the trained model
    model = None
    return model

def predict(model, X):
    # TODO. predict the model
    # should return an array of predictions
    y_pred = np.random.randint(1, 11, len(X))
    return y_pred

if __name__ == '__main__':
    data_df = pd.read_csv('train.csv')
    data_df = data_df.dropna(subset=['OverallRating'])
```

Where the Data Comes From and How It's Used

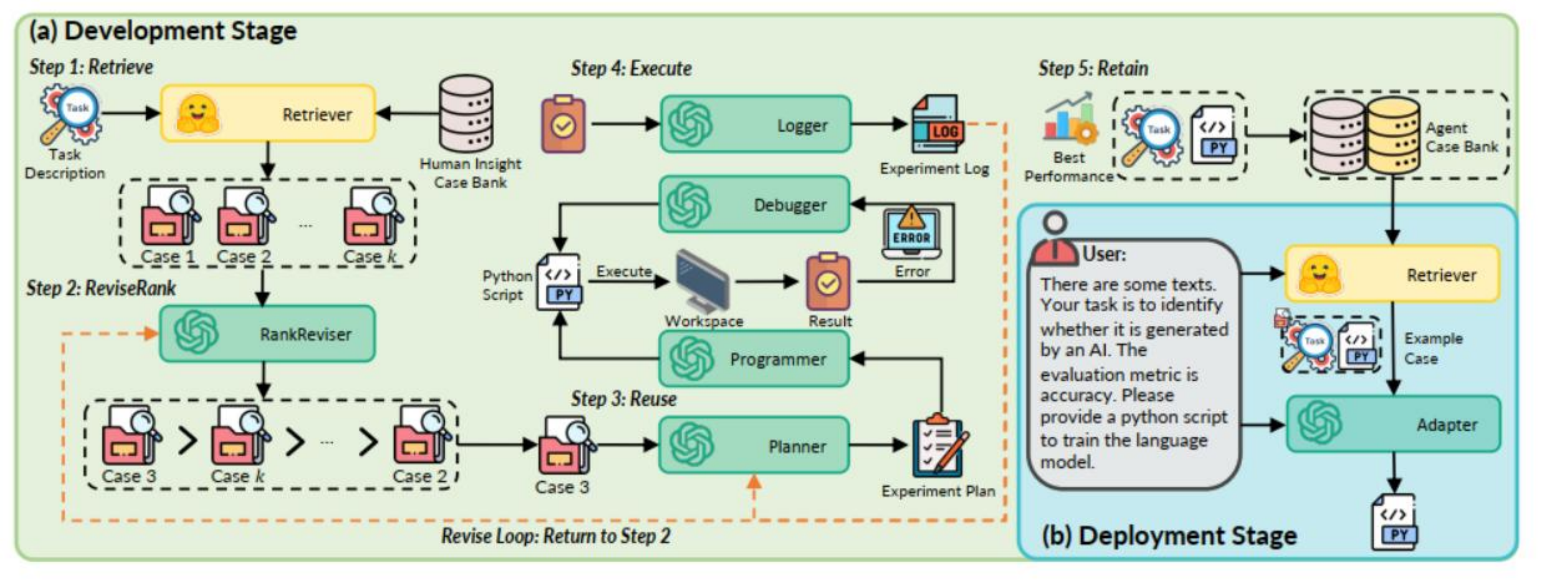
Table 5. Detailed descriptions of selected data science tasks in the experiment.

Stage	Dataset Name	Abbr.	Resource	Modality	Task	Evaluation Metric	Train	Valid	Test
Development	feedback	FB	Kaggle Competition	Text	Regression	MCRMSE	3449	383	79
	airline-reviews	AR	Kaggle Dataset	Text	Regression	RMSE	2997	333	371
	textual-entailment	TE	Kaggle Dataset	Text	Classification	Accuracy	4417	490	4908
	chatgpt-prompt	CP	Kaggle Dataset	Text	Classification	Accuracy	468	116	585
	ett-m2	ETT	Research Dataset	Time Series	Forecasting	MSE	34465	11521	11521
	ili	ILI	Research Dataset	Time Series	Forecasting	MSE	617	74	170
	handwriting	HW	Research Dataset	Time Series	Classification	Accuracy	150	0	850
	ethanol-concentration	EC	Research Dataset	Time Series	Classification	Accuracy	261	0	263
	media-campaign-cost	MCS	Kaggle Competition	Tabular	Regression	RMLSE	291872	32430	324303
	wild-blueberry-yield	WBY	Kaggle Competition	Tabular	Regression	MAE	12384	1376	13761
	spaceship-titanic	ST	Kaggle Competition	Tabular	Classification	Accuracy	6259	695	1739
	enzyme-substrate	ES	Kaggle Competition	Tabular	Classification	AUROC	12019	1335	13355
Deployment	jigsaw	JS	Kaggle Dataset	Text	Regression	RMSE	8639	959	720
	bitcoin-price-prediction	BPP	Kaggle Dataset	Text	Regression	RMSE	1757	195	217
	hotel-reviews	HR	Kaggle Dataset	Text	Regression	RMSE	9220	1024	1025
	webmd-reviews	WR	Kaggle Dataset	Text	Classification	Accuracy	11612	2903	871
	detect-ai-generation	DAG	Kaggle Dataset	Text	Classification	Accuracy	8751	2187	1093
	boolq	BQ	Kaggle Dataset	Text	Classification	Accuracy	1308	327	1635
	traffic	TFC	Research Dataset	Time Series	Forecasting	MSE	12185	1757	3509
	weather	WTH	Research Dataset	Time Series	Forecasting	MSE	36792	5271	10540
	electricity	ELE	Research Dataset	Time Series	Forecasting	MSE	18317	2633	5261
	self-regulation-scp1	SRC	Research Dataset	Time Series	Classification	Accuracy	268	0	293
	uwave-gesture-library	UGL	Research Dataset	Time Series	Classification	Accuracy	120	0	320
	heartbeat	HB	Research Dataset	Time Series	Classification	Accuracy	204	0	250
	crab-age	CA	Kaggle Competition	Tabular	Regression	MAE	59981	6664	66646
	concrete-strength	CS	Kaggle Competition	Tabular	Regression	RMSE	4380	486	4867
	mohs-hardness	MH	Kaggle Competition	Tabular	Regression	MedAE	8430	936	9367
	cirrhosis-outcomes	CO	Kaggle Competition	Tabular	Classification	NLL	6403	711	7115
	smoker-status	SS	Kaggle Competition	Tabular	Classification	AUROC	128997	14333	143331
	software-defects	SD	Kaggle Competition	Tabular	Classification	AUROC	82428	9158	91587

How DS-Agent Works

Process overview

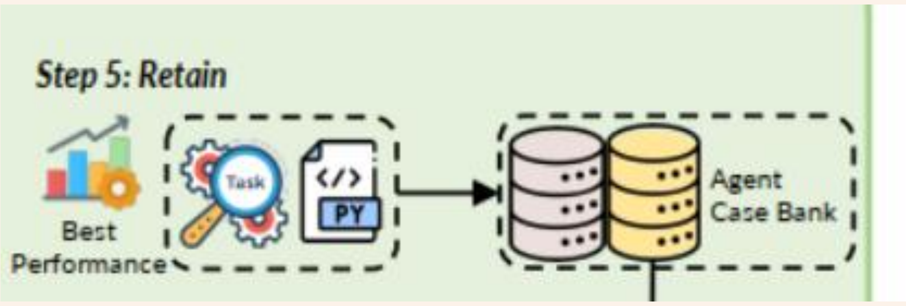
- DS-Agent uses LLMs combined with Case-Based Reasoning (CBR) for better performance.
- Works in two stages: **Development** for refining solutions and **Deployment** for applying them quickly.

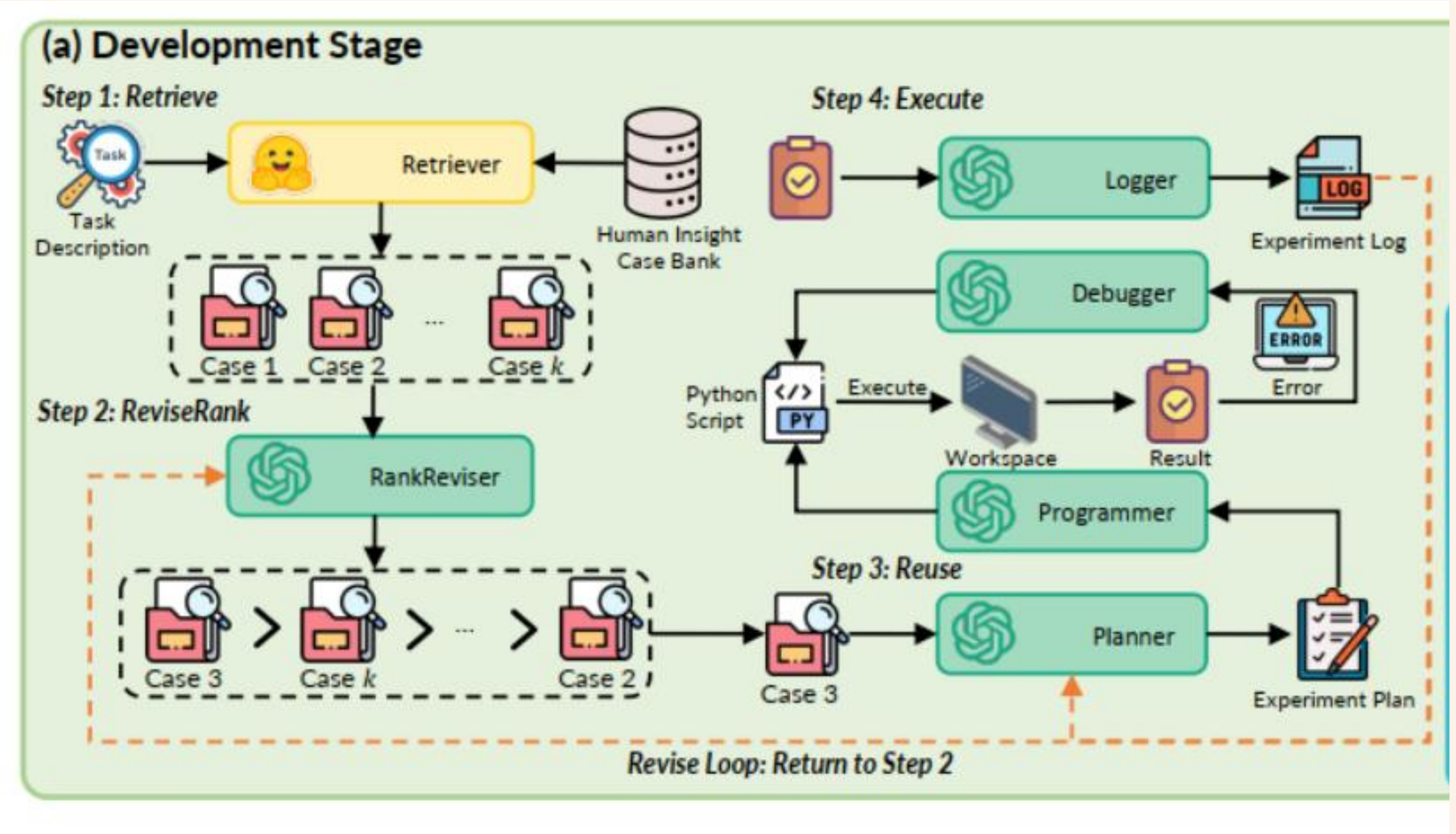


Development Stage Workflow

Five-Step Automatic Iteration Pipeline

- **Retrieve:** Extract relevant cases from the human insight case bank.
- **ReviseRank:** Adjust case rankings using execution feedback.
- **Reuse:** Develop solutions based on retrieved cases.
- **Execute:** Run the experiment using a Python script.
- **Retain:** Archive successful solutions for future reuse.

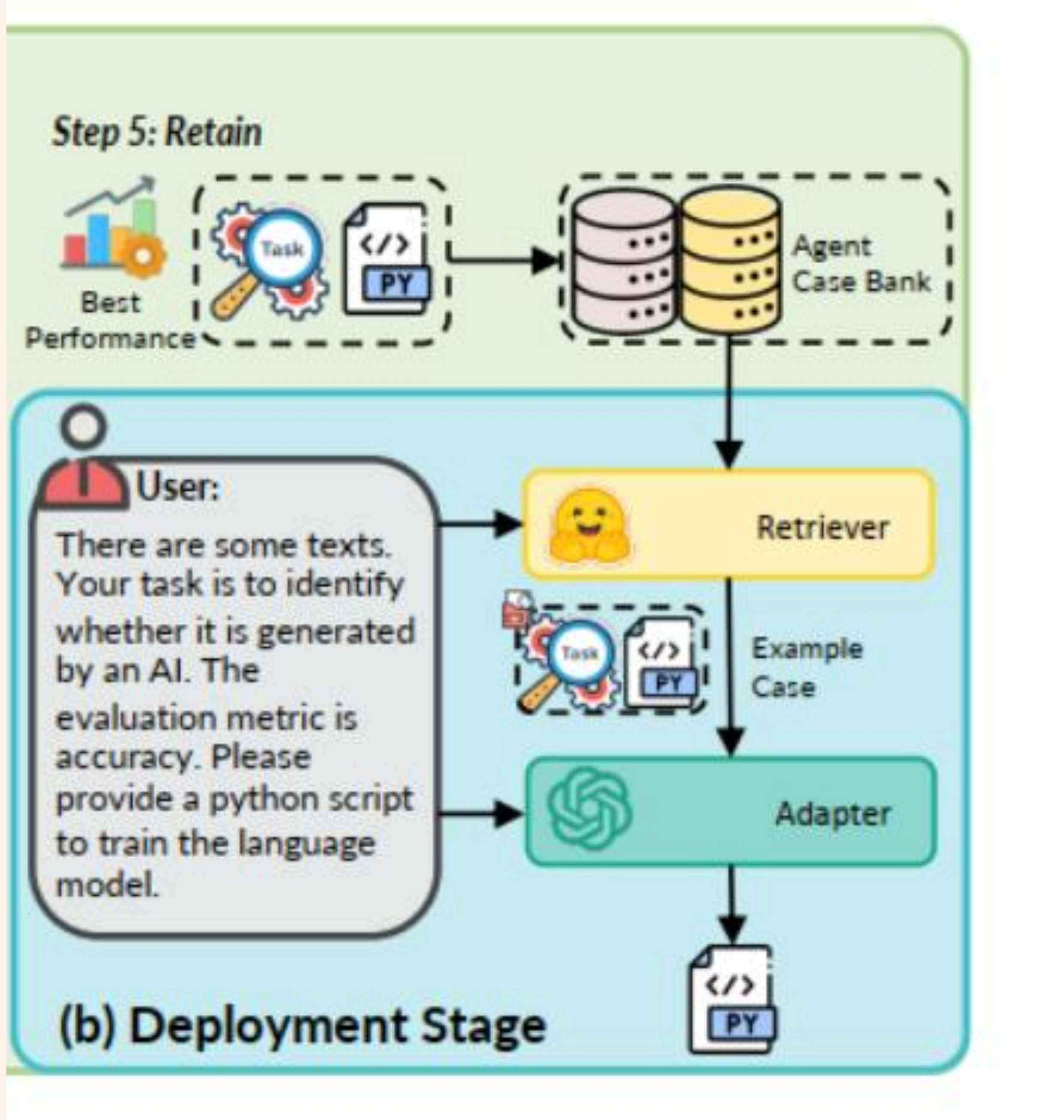




Deployment Stage Workflow

Fast, Resource-Efficient Solution Generation

- Simplifies the process for real-time use by skipping unnecessary iterations.
- Reuses previously successful cases from the agent case bank for current task.
- Focuses on adapting past solutions with minimal modifications.
- Uses fewer resources, making it cost-effective.



DS-Agent's Success Metrics

Model Completion:

- **Development Stage:** Measures success rate—whether the agent can build a bug-free ML model within a set number of steps.
- **Deployment Stage:** Uses the one-pass rate—assesses if the model is successfully built in a single trial.

Model Performance:

- Evaluated using mean rank and best rank to assess the agent’s effectiveness in automated data science tasks.

Resource Cost:

- Tracks how much money is spent when using LLMs to measure efficiency.

DS-Agent Outperforms Across Both Stages

Achieving Higher Success Rates in Development and Deployment

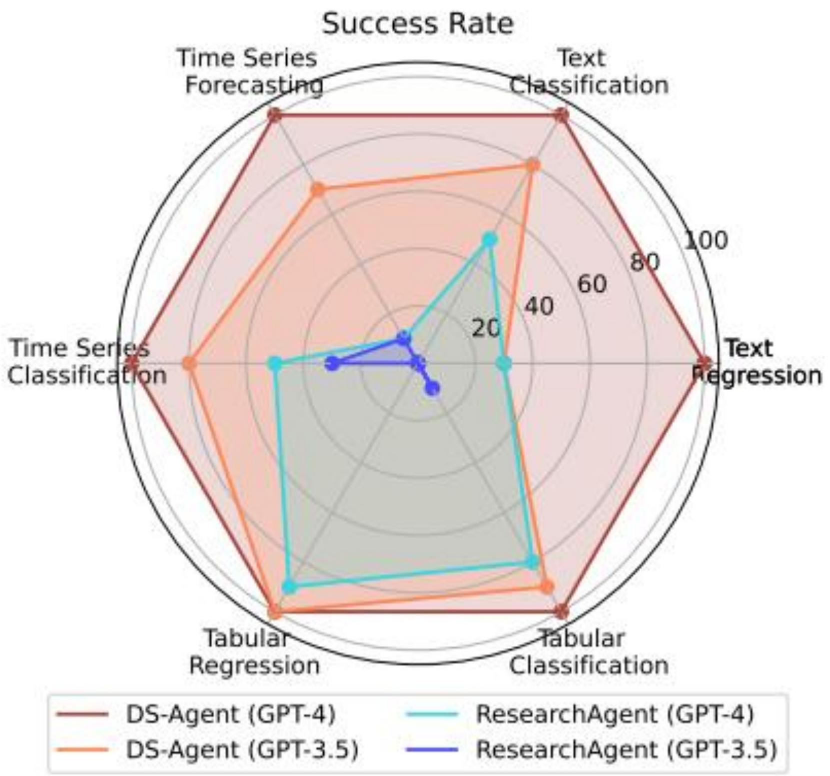


Figure 4. Success rate of four different agents in the development stage. The reported results are averaged across five repetitive trials.

Development Stage

DS-Agent outperforms ResearchAgent using both GPT-3.5 and GPT-4. Achieves a 100% success rate in building ML models across all tasks.

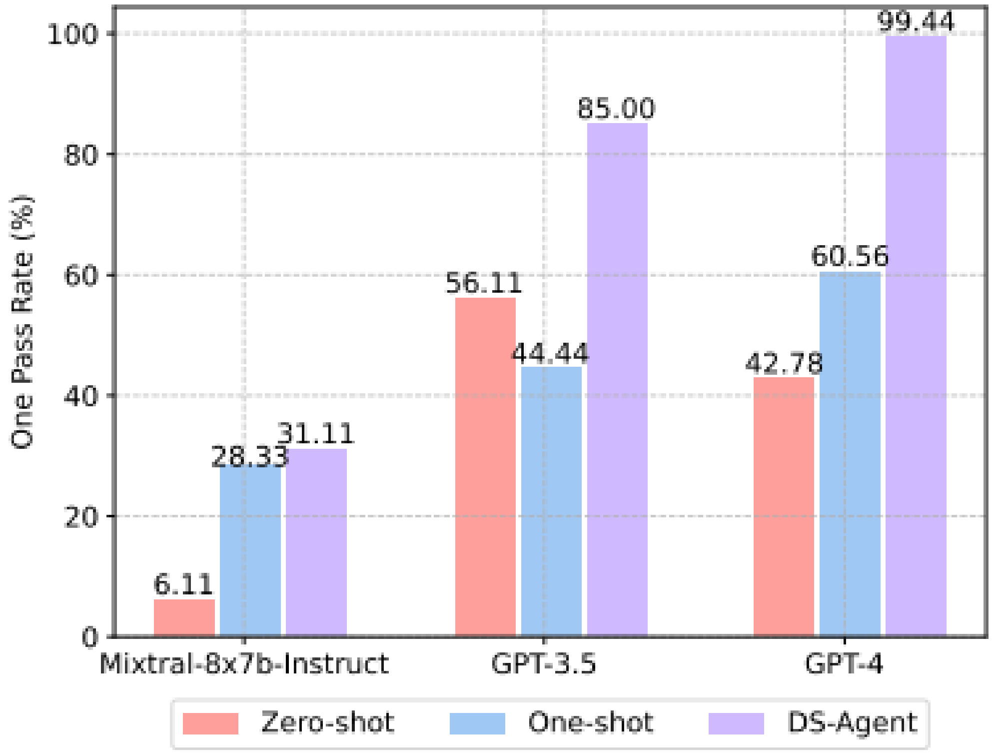


Figure 5. One pass rate of nine different agents over 18 deployment tasks. The reported results are averaged across 10 random runs.

Deployment Stage

DS-Agent surpasses direct prompts across various LLMs. Reaches a 99.4% one-pass rate, demonstrating superior efficiency and accuracy.

DS-Agent Outperforms Across Both Stages

Consistently Higher Rankings in Development and Deployment

Development Stage Performance:

- DS-Agent achieves the lowest mean rank across all tasks compared to ResearchAgent using both GPT-3.5 and GPT-4.
- Achieves the top rank (1.0) in most tasks, demonstrating consistent excellence.

Deployment Stage Performance:

- DS-Agent consistently outperforms zero-shot and one-shot methods across various LLMs.
- Records the lowest mean rank, proving its effectiveness in real-world tasks.

Table 1. Mean rank and best rank w.r.t. task-specific evaluation metric results on 12 data science tasks in the development stage. Results are reported over five repetitive trials. Best performances are highlighted in bold, and second best performances are underlined.

			FB	AR	TE	CP	ETT	ILI	HW	EC	MCS	WBY	ST	ES	Avg
Mean Rank	GPT-3.5	ResearchAgent	8.0	10.0	12.0	13.0	9.4	11.0	14.2	12.2	15.0	16.0	15.8	14.0	12.6
		DS-Agent	7.4	8.2	6.2	7.2	7.2	8.2	6.4	10.2	**6.2**	6.0	7.4	9.6	7.5
	GPT-4	ResearchAgent	7.6	8.6	10.6	11.8	10.0	9.4	12.6	7.2	10.4	10.0	10.6	9.2	9.8
		DS-Agent	**3.4**	**4.2**	**5.8**	**4.4**	**4.4**	**4.4**	**5.4**	**6.6**	6.8	**5.6**	**4.4**	**4.4**	**5.0**
Best Rank	GPT-3.5	ResearchAgent	8.0	10.0	12.0	13.0	7.0	11.0	12.0	9.0	15.0	16.0	15.0	14.0	11.8
		DS-Agent	5.0	2.0	2.0	3.0	3.0	6.0	**1.0**	7.0	2.0	**1.0**	2.0	6.0	3.3
	GPT-4	ResearchAgent	6.0	5.0	7.0	10.0	10.0	3.0	9.0	2.0	**1.0**	2.0	7.0	3.0	5.4
		DS-Agent	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	3.0	**1.0**	4.0	2.0	**1.0**	**1.0**	**1.5**

Table 3. Mean rank w.r.t. task-specific evaluation metric results on 18 data science tasks in the deployment stage. Results are reported over 10 repetitive runs. Best performances are highlighted in bold, and second best performances are underlined.

		JS	HR	BPP	WR	DAG	BQ	TFC	WTH	ELE	SRC	UGL	HB	CA	CS	MH	SS	CO	SD	Avg
Mixtral-8x7b-Instruct	Zero-shot	37.0	35.0	35.0	31.0	35.0	32.0	29.0	32.0	30.0	44.0	54.0	46.0	73.1	66.6	65.8	63.6	33.7	72.0	45.3
	One-shot	35.2	35.0	32.2	31.0	35.0	29.1	29.0	32.0	30.0	36.5	47.1	46.0	50.1	53.1	51.2	51.1	23.6	61.5	39.4
	DS-Agent	37.0	35.0	35.0	31.0	35.0	32.0	29.0	32.0	30.0	20.1	16.4	38.5	25.3	54.5	53.7	53.9	32.2	47.6	35.5
GPT-3.5	Zero-shot	21.7	35.0	30.1	28.6	27.1	28.3	27.1	29.1	28.1	33.1	48.4	21.4	29.0	35.3	28.8	35.7	25.2	42.3	30.8
	One-shot	27.6	25.8	27.6	25.6	34.6	23.0	20.8	29.1	27.0	35.7	48.4	**21.1**	27.1	50.5	58.4	57.5	33.9	56.4	35.0
	DS-Agent	**6.0**	22.6	15.0	20.6	15.1	**13.1**	17.3	13.4	14.4	**20.0**	13.0	23.0	29.0	19.3	**7.6**	**2.0**	37.0	19.5	17.1
GPT-4	Zero-shot	36.7	31.8	35.0	29.0	29.4	32.0	29.0	32.0	30.0	37.3	45.7	33.6	**1.0**	**15.3**	23.2	17.9	28.3	20.1	28.2
	One-shot	35.1	24.4	**13.8**	26.6	29.6	28.8	23.1	30.1	26.6	26.7	41.6	36.7	29.7	21.9	35.3	28.9	21.4	23.2	28.0
	DS-Agent	18.6	**1.0**	14.6	**5.2**	**6.2**	18.8	**15.7**	**6.3**	**8.1**	**20.0**	**11.4**	21.2	**1.0**	32.6	14.5	8.2	**13.0**	**12.4**	**12.7**

DS-Agent's Success Metrics

Efficient Cost Reduction Through Two-Stage Design

Two-Stage Framework:

- The development stage focuses on creating effective model designs, requiring more resources.
- The deployment stage efficiently solves tasks with minimal resources.

Significant Cost Reduction:

- DS-Agent cuts costs by over 90% in the deployment stage compared to the development stage.
- A single run costs only $0.0045 with GPT-3.5 and $0.1350 with GPT-4 during deployment.

Real-World Advantage:

- Low costs and high efficiency make DS-Agent an ideal solution for large-scale or resource-constrained deployments.

Table 4. Monetary cost comparison among development and deployment stage on a single run.

DS-Agent	Development Stage	Deployment Stage	Cost Deduction Percentage
GPT-3.5	$0.06	$0.0045	92.5%
GPT-4	$1.60	$0.1350	91.5%

Pros and Cons of DS-Agent

Advantages:

- Highly efficient in using resources and achieving high success rates.
- Flexible, adaptable, and scalable across different data science projects.
- Makes great use of expert knowledge from past successful cases.

Disadvantages:

- Needs a large, high-quality case database for top performance.
- Can struggle with completely new problems without similar past examples.

Future Improvements

Balancing Automation with Expert Oversight

- **Human-Guided Experiment Planning**
  - Data Scientist experts define task objectives, data selection, the data cleaning techniques and suggest baseline models or feature engineering strategies.
  - DS-Agent automates data processing, model selection, and tuning.

- **Customizable AI Assistance**
  - Users set automation levels (full auto vs. human-in-the-loop).
  - Choose between speed, accuracy, or interpretability.

University of Central Florida

Machine Learning for Biomedical Data

THANK YOU