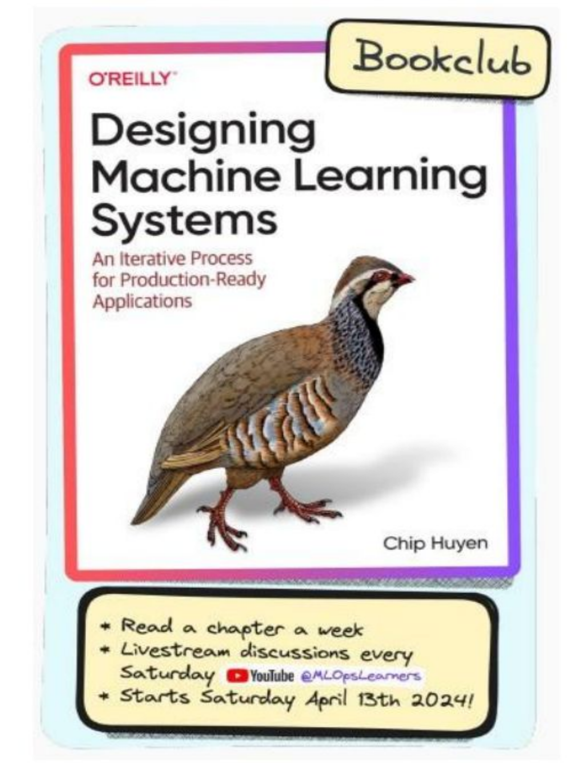


Bookclub

O'REILLY

Designing Machine Learning Systems

An Iterative Process for Production-Ready Applications

Chip Huyen

- Read a chapter a week
- Livestream discussions every Saturday YouTube @MLOpsLearners
- Starts Saturday April 13th 2024!

Chapter 9: Continual Learning and Test in Production

Group 8 (ML Mavericks)

Saturday, June 8, 2024

YouTube

https://www.youtube.com/watch?v=pVUav52InoY&t=1s

Chapter 9: Continual Learning & Test in Production

Designing Machine Learning Systems

ML Mavericks

Vishnu Vardhan

LinkedIn

Kaushal Prajapati

LinkedIn

Shivalika Singh

LinkedIn

Group 8

Agenda

- Continual learning
  - Scenarios
  - Techniques
  - Implementation
  - Changes with GenAI for CL
- Test in Production
  - Techniques:
    - Shadow deployment, A/B testing, Canary release, Interleaving experiments, Bandits
  - Challenges with GenAI
  - Approaches for GenAI

Continual Learning

> After training on a non-stationary objective, neural networks exhibit a reduced ability to adapt to new tasks. This loss of plasticity occurs most robustly when the relationship between inputs and prediction targets changes over time, and the network must learn to ‘**overwrite**’ its prior predictions
>
> – Clare Lyle, Google DeepMind

https://arxiv.org/pdf/2303.01486 – understanding plasticity in NN

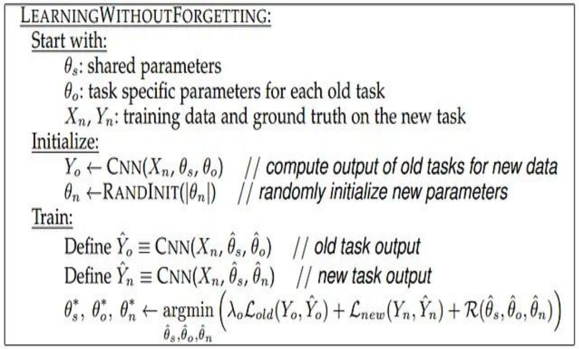
Three scenarios for continual learning

- **Domain Incremental (DI) continual learning** comprises all cases where data distribution changes over time. (Data iteration)
- **Task Incremental (TI) continual learning** is classic multi-task learning but in an incremental way. (Model iteration)
- **Class Incremental (CI) continual learning** is a scenario in which the number of classes in a classification task is not fixed but can increase over time. (Model Iteration | & Data Iteration)

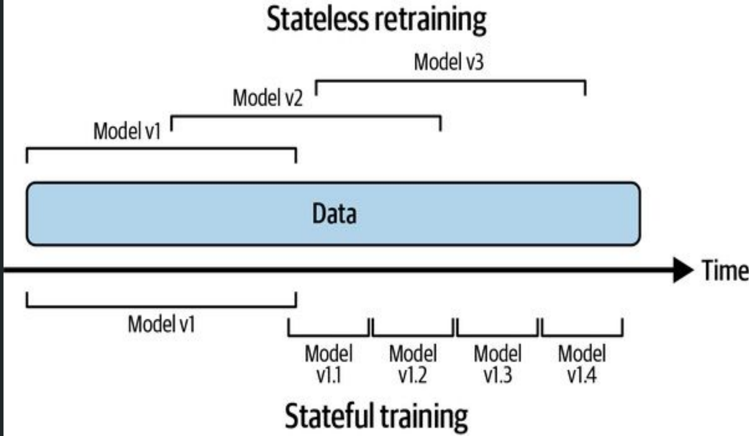
https://arxiv.org/pdf/1904.07734 - 3 scenarios of CL

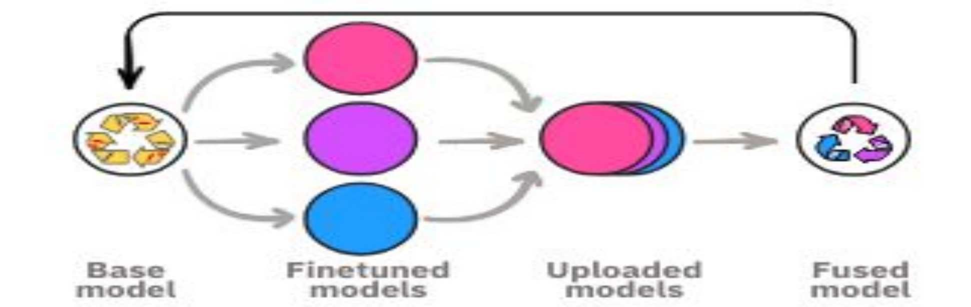
Techniques for continual learning

- Memory (Rehearsal)
    - Stateless
    - Stateful
- Regularization
    - EWC
    - LwF
- Architectural
    - Surgery Transfers
    - ColD Fusion (for TI)
    - Differential LR
    - Same backbone multiple heads
    - Adapters



```
LEARNINGWITHOUTFORGETTING:
Start with:
    $\theta_s$: shared parameters
    $\theta_o$: task specific parameters for each old task
    $X_n, Y_n$: training data and ground truth on the new task
Initialize:
    $Y_o \leftarrow \text{CNN}(X_n, \theta_s, \theta_o)$    // compute output of old tasks for new data
    $\theta_n \leftarrow \text{RANDINIT}(|\theta_n|)$    // randomly initialize new parameters
Train:
    Define $\hat{Y}_o \equiv \text{CNN}(X_n, \hat{\theta}_s, \hat{\theta}_o)$    // old task output
    Define $\hat{Y}_n \equiv \text{CNN}(X_n, \hat{\theta}_s, \hat{\theta}_n)$    // new task output
    $\theta_s^*, \theta_o^*, \theta_n^* \leftarrow \operatorname*{argmin}_{\hat{\theta}_s, \hat{\theta}_o, \hat{\theta}_n} \left(\lambda_o \mathcal{L}_{old}(Y_o, \hat{Y}_o) + \mathcal{L}_{new}(Y_n, \hat{Y}_n) + \mathcal{R}(\hat{\theta}_s, \hat{\theta}_o, \hat{\theta}_n)\right)$
```





https://arxiv.org/pdf/1909.08383 Survey in CL

Implementation

- Is CL required ?
  - Manual -> Stateless (scratch, cron) -> Stateful (fixed time interval) -> CL
- Plasticity - Stability, what’s priority ?
- Rare events , Cold Start Problem ?
- Can we divide and conquer ?
- How much can you spend ? Challenges
  - Is there a data distribution shift ? Is it significant ?
  - How early can you ingest data ? access to fresh data
  - How labels ? label hunting
  - Is data reliable ? adversarial attacks
  - How often to update ? value vs cost
  - Is the update, good enough ?

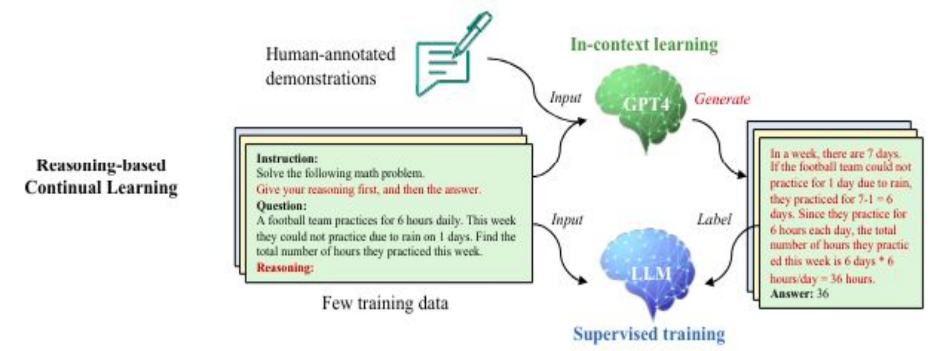
Tips

- Memory based
  - most effective but expensive too, plasticity is critical
- Architecture based
  - Easiest and practical for transformers based models
- Regularisation based
  - Easy, stability is critical
- Combination of above 3
- Architecture matters
  - Bigger the model higher chances of generalization to multiple tasks
  - Representation Learning lowers forgetting
  - BN, MaxPool, Wider networks helps in CL, Increasing no of heads in ViT lowers forgetting

https://arxiv.org/pdf/2202.00275 - Architecture matters in CL

Gen AI

- With a massive MoE which is great at reasoning all you need is “right context”
- Incontext learning & Emergent abilities
  - Prompt engineering + Few shot
  - Problem of Sycophancy
- Domain adaptation
  - Continual Pretraining
  - LoRA, QLoRA, DoRA, RSLoRA
  - LoftQ, PiSSA init
  - Differential LR, LoRA+
  - Model Merging - SLERP, TIES, LERP
- (Prefix, Prompt, Soft) tuning methods
- TRACE - benchmark for CL in LLM, RCL



Instruction:
Solve the following math problem.
Give your reasoning first, and then the answer.
Question:
A football team practices for 6 hours daily. This week they could not practice due to rain on 1 days. Find the total number of hours they practiced this week.
Reasoning:

In a week, there are 7 days. If the football team could not practice for 1 day due to rain, they practiced for 7-1 = 6 days. Since they practice for 6 hours each day, the total number of hours they practiced this week is 6 days * 6 hours/day = 36 hours.
Answer: 36

https://arxiv.org/abs/2405.09673 - lora learns less, forget less

https://unsloth.ai/blog/contpretraining - improvement on above

The biggest challenge of continual learning isn’t in writing a function to continually update your model—you can do that by writing a script!

The biggest challenge is in making sure that this update is good enough to be deployed.

So let’s learn about, how to **Test in Production.**

Test in Production - Techniques

Shadow Deployment

Process:

- Deploy candidate model with existing model
- Route requests to both models
- Serve predictions of only existing model
- Monitor & analyze the predictions of candidate model
- Replace existing model with new model once satisfied with new model's performance.

Advantage:

- Safest approach – no risk from deploying new model since its not served to users.

Disadvantage:

- It's expensive – doubles inference cost.

A/B Testing

Process:

- Deploy candidate model in parallel with existing model
- Route a small percentage of the traffic to new model & the rest is routed to the existing model.
- Monitor and analyze the predictions of both models.
- Check if the performance difference b/w both models is statistically significant.

Key Points:

- Predictions of one model shouldn't be influencing the predictions of another one.
- Traffic to both models should be truly random.
- Test with sufficient no. of samples.

Canary Release

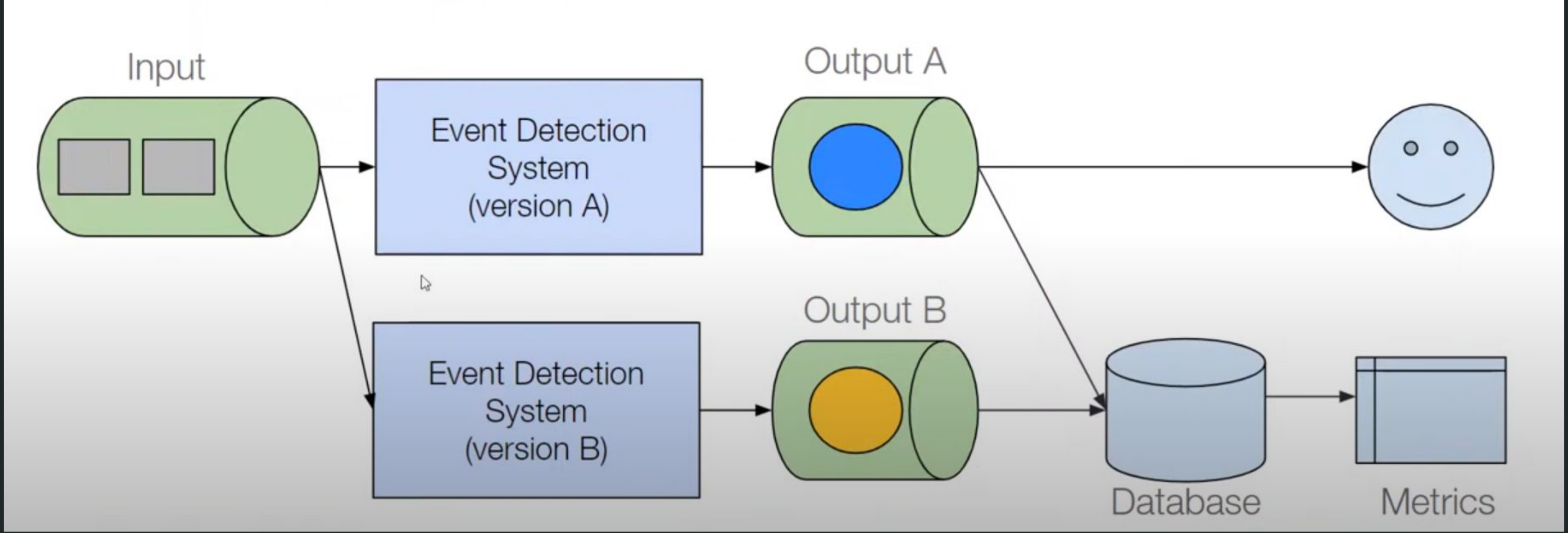
Process:

- Deploy candidate (canary) model with existing model
- Route a portion of the traffic to candidate model.
- Measure the candidate model's performance against the existing model's metrics.
- Check performance of candidate model. If satisfactory, increase traffic otherwise abort.
- Stop when the canary serves all traffic (candidate model replaces existing model) or when the canary is aborted.

Key Points:

- Canary releases can implement A/B testing setups.
- Can be rolled out to less critical markets initially before rolling out to everybody.

Case Study: Real Time Event Detection System - Shadow Stack Deployment


Input
Event Detection System (version A)
Output A
Event Detection System (version B)
Output B
Database
Metrics

Reference Link

Test in Production - Techniques

Interleaving Experiments

- **Concept:**
  - Users are exposed to recommendations from multiple models simultaneously to see which one performs better.
- **Advantage:**
  - Identifies the best algorithms with a smaller sample size compared to A/B testing.
- **Implementation:**
  - Use methods like team-draft interleaving to ensure recommendations from models A and B are equally likely at any position.

Bandits

- **Principle:** Balance between exploitation (using the best-known option) and exploration (trying new options).
- **Application in ML:** Evaluate multiple models as slot machines to maximize prediction accuracy while determining the best model.
- **Algorithms:**
  - **Epsilon-greedy:** 90% traffic to the best-performing model, 10% to a random model.
  - **Thompson Sampling:** Selects models based on the probability of optimal performance.
  - **Upper Confidence Bound (UCB):** Chooses models with the highest upper confidence bound, factoring in uncertainty.

Challenges:

- Bandits are more complex to implement than A/B testing but are more data-efficient and reduce opportunity costs.
- Used primarily by big tech companies due to implementation complexity

Gen AI - Challenges to Test in Production

Key Challenges:

1. **Established Metrics for Business Use Cases:**
   - Define specific metrics tailored to the business use case of the deployed LLM.
   - Public benchmarks like MMLU are too generic and may not align with specific business needs.
2. **Task-Specific Evaluation:**
   - LLMs can perform various tasks: QA, coding, content analysis, content generation, summarization, etc.
   - Each task requires distinct evaluation metrics and pipelines tailored to its requirements.
3. **Automation of Evaluation Pipelines:**
   - Manual evaluations are time-consuming and inconsistent.
   - Automating the evaluation process is critical for scalable, efficient, and objective testing in production.
4. **Safety Testing:**
   - Continuous safety testing is crucial, especially when LLM responses affect people and society directly.
   - Ensure the LLM does not have inherent biases, is not influenced by new trends, and avoids issues like prompt injection.
   - Regularly update safety protocols to address emerging biases and societal impacts.

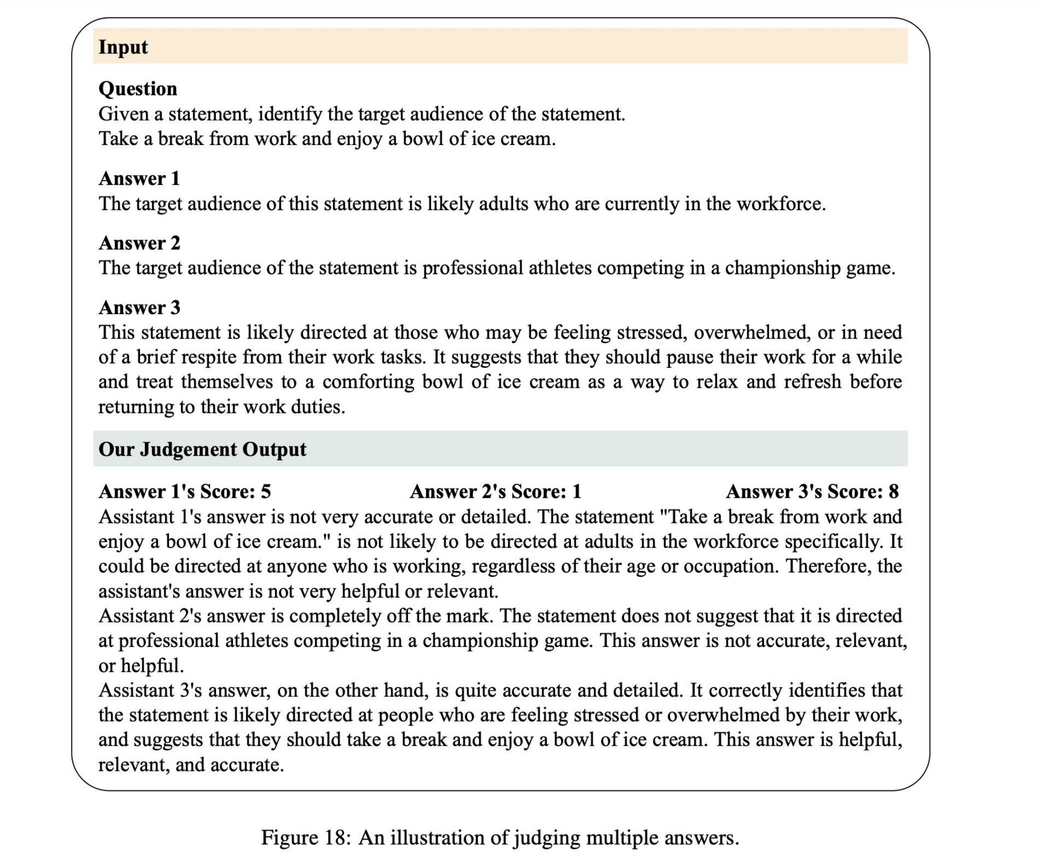
***Possible* Solution:**

- **Automated Evaluation with LLMs:** Ensures robust, scalable evaluations and facilitates continual learning and online testing.

JudgeLM paper

JudgeLM is a scalable language model judge, designed for evaluating LLMs in open-ended scenarios. It achieves an agreement exceeding 90% that surpasses the human-to-human agreement.

- Grading a single answer
- Judging multiple answers
- Judging multimodal models
- Judging multi turn chat



Input

Question
Given a statement, identify the target audience of the statement.
Take a break from work and enjoy a bowl of ice cream.

Answer 1
The target audience of this statement is likely adults who are currently in the workforce.

Answer 2
The target audience of the statement is professional athletes competing in a championship game.

Answer 3
This statement is likely directed at those who may be feeling stressed, overwhelmed, or in need of a brief respite from their work tasks. It suggests that they should pause their work for a while and treat themselves to a comforting bowl of ice cream as a way to relax and refresh before returning to their work duties.

Our Judgement Output

Answer 1's Score: 5 **Answer 2's Score: 1** **Answer 3's Score: 8**
Assistant 1's answer is not very accurate or detailed. The statement "Take a break from work and enjoy a bowl of ice cream." is not likely to be directed at adults in the workforce specifically. It could be directed at anyone who is working, regardless of their age or occupation. Therefore, the assistant's answer is not very helpful or relevant.
Assistant 2's answer is completely off the mark. The statement does not suggest that it is directed at professional athletes competing in a championship game. This answer is not accurate, relevant, or helpful.
Assistant 3's answer, on the other hand, is quite accurate and detailed. It correctly identifies that the statement is likely directed at people who are feeling stressed or overwhelmed by their work, and suggests that they should take a break and enjoy a bowl of ice cream. This answer is helpful, relevant, and accurate.

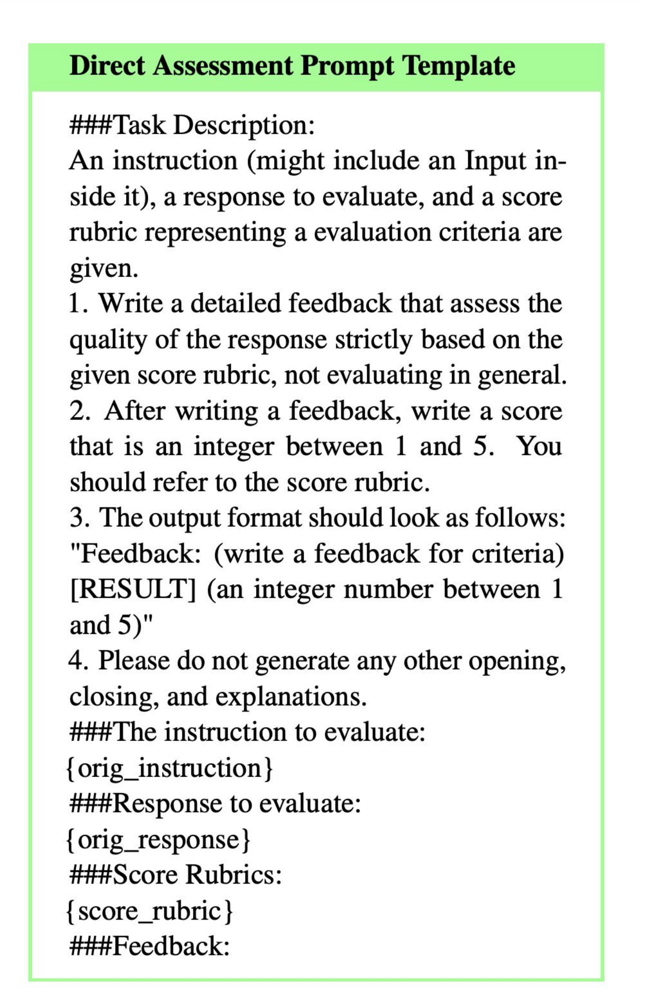
Figure 18: An illustration of judging multiple answers.

Prometheus paper

Prometheus is a family of open-source language models specialized in evaluating other language models.

The Prometheus 2 models support both direct assessment (absolute grading) and pairwise ranking (relative grading).

You could switch modes by providing a different input prompt format and system prompt. Within the prompt, you should fill in the instruction, response(s), and score rubrics with your own data. Optionally, you could also add a reference answer which leads to better performance!



Direct Assessment Prompt Template

###Task Description:
An instruction (might include an Input inside it), a response to evaluate, and a score rubric representing a evaluation criteria are given.
1. Write a detailed feedback that assess the quality of the response strictly based on the given score rubric, not evaluating in general.
2. After writing a feedback, write a score that is an integer between 1 and 5. You should refer to the score rubric.
3. The output format should look as follows: "Feedback: (write a feedback for criteria) [RESULT] (an integer number between 1 and 5)"
4. Please do not generate any other opening, closing, and explanations.
###The instruction to evaluate:
{orig_instruction}
###Response to evaluate:
{orig_response}
###Score Rubrics:
{score_rubric}
###Feedback:

Discussion Questions

- In CI advancement journey at which stage your company is ?
  - Manual, Stateless, Stateful, Fully Automated
- Are are working with self hosted, open source LLMs ? How you handle domain adaptation, data shifts ?
- In your experience, what factors did you consider before choosing a testing technique for your use case?
- Did you find yourself switching from testing approach to another in production for your use case? What challenges did you face or what reasons motivated you to switch ?

References

- https://arxiv.org/pdf/2203.13381 - probing representation forgetting
- https://arxiv.org/pdf/2303.01486 - understanding plasticity in NN
- https://arxiv.org/pdf/1904.07734 - 3 scenarios of CL
- https://arxiv.org/pdf/2202.00275 - Architecture matters in CL
- https://arxiv.org/abs/1612.00796 - Overcoming forgetting
- https://arxiv.org/abs/1606.04671 - Progression NN
- https://scholar.google.fr/citations?hl=en&user=YLh7yrwAAAAJ&view_op=list_works - Rahaf Aljundi blogs and talks
- https://arxiv.org/pdf/1909.08383 - survey on CL
- https://blog.arcee.ai/case-study-innovating-domain-adaptation-through-continual-pre-training-and-model-merging - continual pretraining and model merging for domain adaptaion
- https://arxiv.org/abs/2405.09673 - lora learns less, forget less
- https://unsloth.ai/blog/contpretraining - unsloth, showing improvements over the above paper
- https://openreview.net/pdf?id=xelrLobW0n - TRACE