Aalto University
School of Science

Automatic Prompt Engineering Pipeline for LLM Applications

Enhancing LLM Performance through Continuous Monitoring and Feedback-Based Optimization

Federico Rubbi – Project Presentation

Advanced Topics in Software Systems – a.y. 2024-2025

Project Overview

Project Overview

Introduction

This project aims to develop a system that optimizes language model prompts automatically. With the aid of monitoring tools and advanced search algorithms, the system iteratively improves LLM performance, ensuring high-quality responses with minimal manual intervention.

Project Overview

Challenge

Optimizing the behavior of large language models (LLMs) is difficult. Key issues include:

- Performance Variability: Even small prompt adjustments can drastically change outputs.
- Manual Tuning: Current prompt tuning requires extensive trial and error, which is time-consuming.
- Quality Assurance: Ensuring that responses meet quality, ethical, and cost-effectiveness standards.

Project Overview

Goal

The objective is creating a self-optimizing system that continuously improves prompts for LLMs, using a data-driven approach to enhance response quality, relevance, and reliability without manual prompt engineering.

Project Overview

Study Questions

1. How effectively can an optimization algorithm improve prompt quality for LLMs over time?

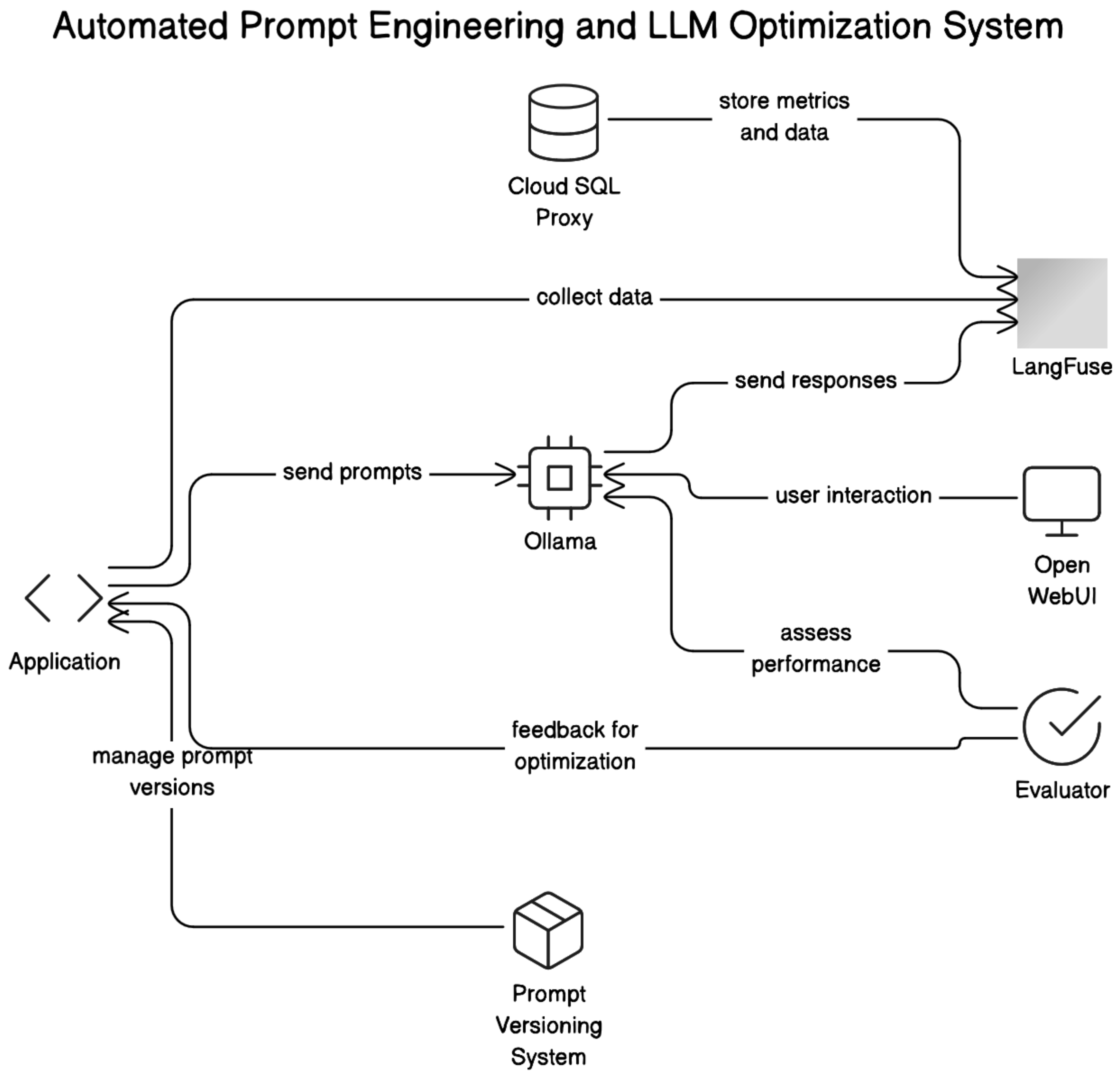
2. What metrics are most influential in evaluating and optimizing LLM prompts?

3. What is the impact of user feedback on iterative prompt optimization?

Proposed Solution

Proposed Solution

Architecture Overview



Proposed Solution

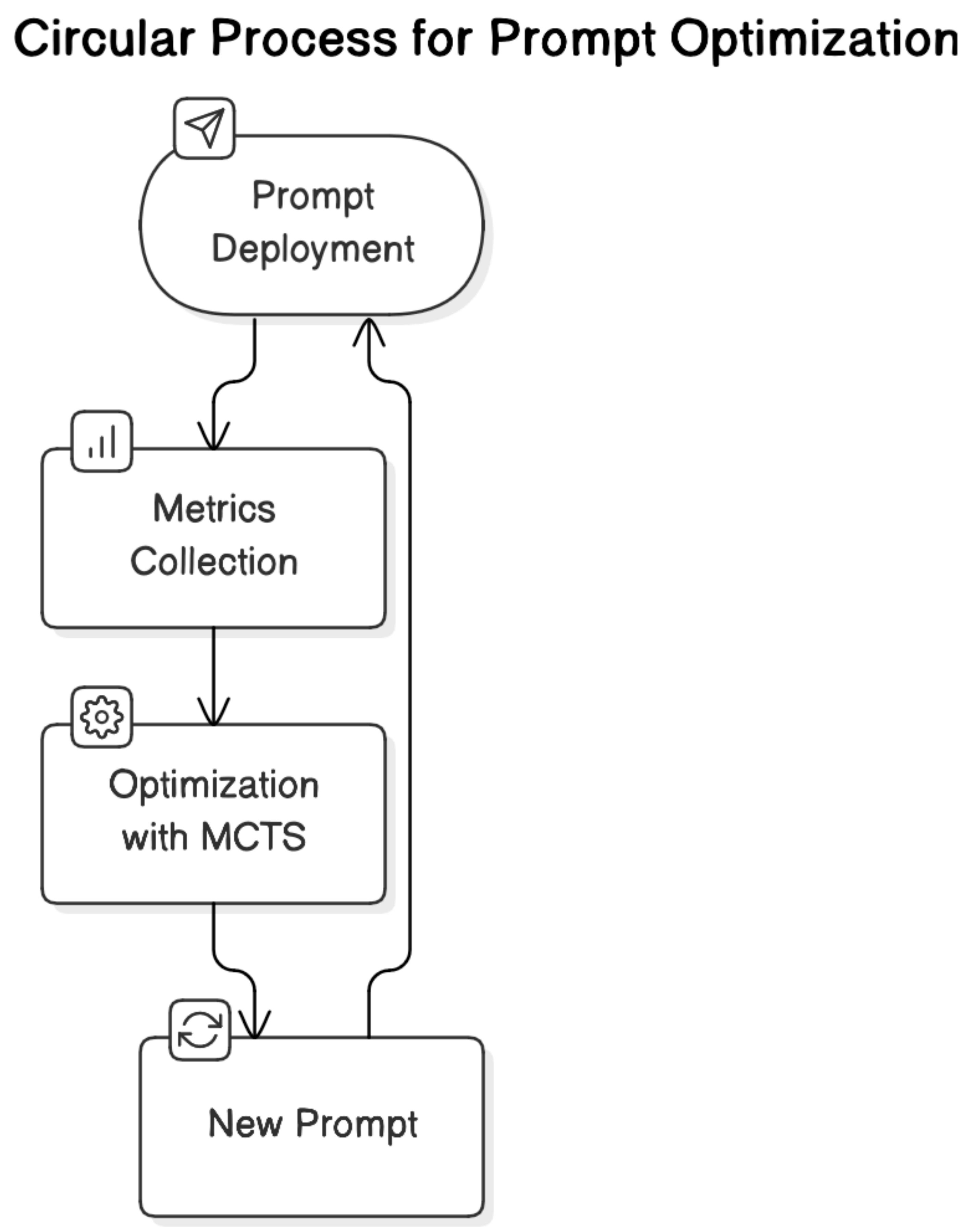
Key Components

1. **LangFuse**: A monitoring tool to track and analyze key metrics (like relevance, quality, and cost).

2. **Optimizer**: An optimization algorithm to explore different prompt variations, finding the best-performing options. A candidate algorithm for the optimization is Monte Carlo Tree Search (MCTS).

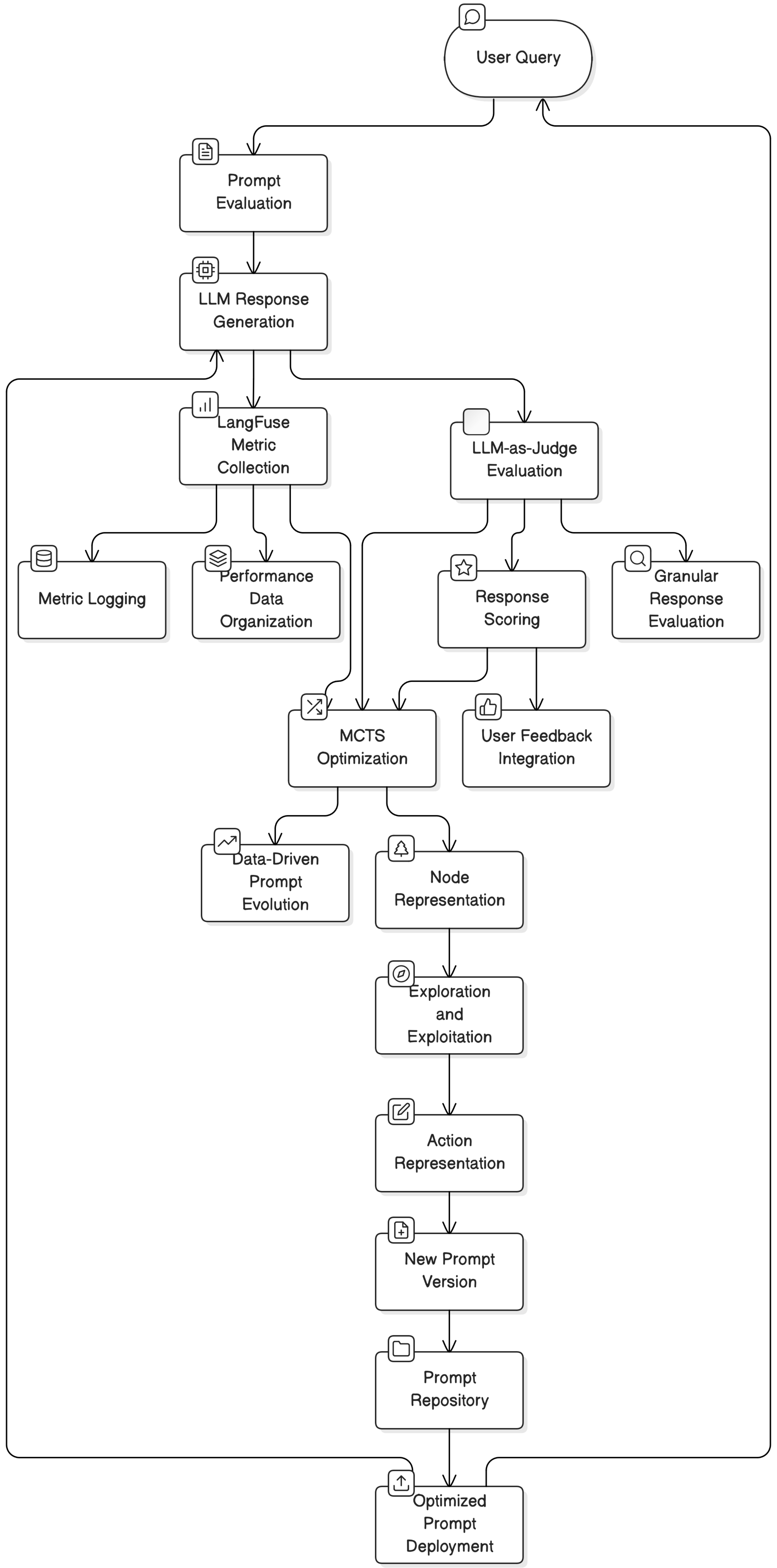
3. **Feedback Loop**: The system continuously deploys new prompts, collects metrics, and iteratively refines prompts to improve performance.

Proposed Solution
Key Components



Proposed Solution

System Design



Optimization Process

Optimization Process

MCTS Algorithm Steps

1. **Selection**: Traverse the tree from the root, selecting child nodes based on a balance of exploration (unvisited nodes) and exploitation (high-reward nodes).

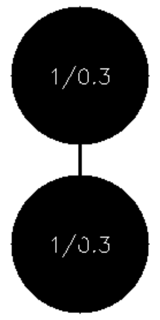
2. **Expansion**: Expand the tree by adding a new child node if the selected node is not terminal.

3. **Simulation**: Perform a simulated random play-out (or evaluation) from the new node to estimate the outcome.

4. **Backpropagation**: Update the path of nodes from the leaf to the root with the simulation results, adjusting each node’s value based on the outcome.

Optimization Process

MCTS Visualization



MCTS algorithm visualization. Nodes of the trees report visit found and cumulative value.

Optimization Process

Upper Confidence Bound for Trees

$$\mathrm{UCT}(\mathrm{node}_i) = v_i + C \cdot \sqrt{\frac{\ln N_i}{n_i}}$$

UCT formula [Kocsis and Szepesvári, 2006] involving the ratio between the logarithm of the number of visit to the parent node and the number of visits to the children nodes.

Technologies and Tools

Technologies and Tools

LangFuse

- Used to monitor interactions with the LLM and collect both intrinsic metrics (e.g., perplexity, latency, cost) and custom metrics (e.g., toxicity, relevance).
- Provides real-time observability and insights into LLM behavior.

Technologies and Tools

LLM Deployment

- The LLM is deployed locally and interacts with prompts through an API, generating responses based on the latest prompt variations.

Technologies and Tools

Prompt Repository

- Storage Solution: Uses a version-controlled repository (e.g., Git) to maintain all prompt modifications.
- Historical Tracking: Allows for easy rollback and comparison of different prompts, creating a clear history of changes.

Technologies and Tools

Prompt Modification Module

- Dynamic Adjustments: Requests prompt adjustments from a Judge LLM based on recent metrics.
- Guided Suggestions: Based on the latest metrics, it suggests modifications, like small line adjustments, to refine performance.

Concluding considerations

Concluding considerations

Benefits of the System

- **Automated Optimization**: The system reduces the need for manual prompt tuning, using data-driven adjustments for continuous improvement.
- **Enhanced Response Quality**: Through a structured feedback loop, the LLM adapts to deliver increasingly relevant, coherent, and ethical responses.
- **Scalability and Adaptability**: The architecture can support various LLMs and be customized for different applications by adjusting metrics and feedback mechanisms.