Mrs. Pratiksha Shevatekar, 2026, 14:2
ISSN (Online): 2348-4098
ISSN (Print): 2395-4752

International Journal of Science, Engineering and Technology

An Open Access Journal

A Hybrid AI and User-Driven Approach for Prompt Evaluation and Ranking in Large Language Models

Mrs. Pratiksha Shevatekar, Mrs. Pooja Mishra,Prasad Patel, Raghav Tandulkar, Lokesh Dhoble, Sairaj Patil

Associate Professoer and UG schoolars,
Dept. of Computer Engineering,
Dr. D. Y. Patil Institute of Engineering, Management and Research
Pune, Maharashtra

Abstract- Unexpected progress in smart programs now shapes daily work in writing, programming, code fixes. As big language tools appear, old ways shift fast - yet better results do not come automatically. Often, flaws stem less from tech limits than weak human requests. Unclear questions tend to produce mismatched answers, raising workload while slowing output. While some sites allow sharing those queries, few offer solid ways to judge their strength, leaving users sifting through clutter where most entries add little value. Facing these issues, PromptX applies a structured method meant to improve how prompts work in practice. Through automatic assessments paired with insights from users, it judges what truly performs well. Instead of favoring only widely used options, relevance within context takes priority. As selections adapt gradually, less useful repetitions fade out naturally. Suggestions shift quietly based on individual habits and changing needs. Performance stays stable whether crafting text, solving problems, or generating code. With repeated use, refinements emerge without interruption. The system evolves simply because it is used.

Keywords— Prompt Engineering, Large Language Models, AI Evaluation, Prompt Ranking, Recommendation Systems, Prompt Marketplace.

I. INTRODUCTION

These days, tools like ChatGPT, DALL·E, and Midjourney shift how individuals write, code, or manage intricate work. Since performance depends on what users provide, the way requests are built affects outcomes noticeably. Well-structured directions tend to generate useful replies; unclear ones usually bring erratic answers instead. When key parts - like background, goal, or limits - are absent, trust in results drops sharply. So it happens that clear prompts now matter greatly when people work alongside AI systems. Still, even with smarter machines, plenty of individuals find it hard to state what they want. Vague terms, missing details, or questions too wide in scope tend to damage outcomes, pushing people into loops of testing and fixing, which takes time and adds frustration. A tiny flaw in how a request is built might spread across answers, showing why better organization in asking could help.

Still unclear because direction and assessment tools remain missing. Even when certain sites gather prompts, dependable responses rarely follow, so judging value becomes hard. Old or weak examples pile up, confusion grows. Without scores, comments, or clear signs of success, trust stays low. A different path emerges through PromptX: a space built for testing, sorting, refining how prompts work. Quality checks happen automatically at the same time real input shapes results - each judged by precision,

usefulness, strength. Gradually adjusting to how people interact, it shapes suggestions based on patterns seen during regular use. With a clear framework guiding responses, guesswork fades while reliability grows stronger through repeated application. Support remains steady whether assisting with writing code, generating text, or streamlining routine actions.

II. LITERATURE SURVEY

TABLE 1

No.	Paper Title	Journal	Author & Year	Methodology
1	From Prompts to Performance: Leveraging LLMs for Enhanced Educational AI Interactions.	IEEE	Bhatt et al., 2025	Empirical analysis of prompt-response pairs using Likert scale and statistical correlation analysis
2.	Large Language Models Are Human-Level Prompt Engineers (Automatic Prompt Engineer, APE)	arXiv	Zhou et al., 2022	AI-based prompt generation and scoring using task performance evaluation metrics
3.	Prompting Frameworks for Large Language Models: A Survey	arXiv	Liu et al., 2023	Survey of prompt frameworks, lifecycle models, and system design approaches
4.	Chatbot Arena: An Open Platform for Evaluating LLMs by Human Preference	arXiv	Chiang et al., 2024	Human preference-based ranking using pairwise comparisons and statistical ranking models
5.	The Promises and Pitfalls of LLMs as Feedback Providers (Teacher Education Prompt Study)	Journal	Jacobsen & Weber, 2025	Rubric-based prompt evaluation comparing LLM, novice, and expert feedback outputs
6.	Unleashing the Potential of Prompt Engineering for Large Language Models (Survey)	Patterns	Chen et al., 2023	Review of advanced prompt techniques including CoT, RAG, and evaluation strategies
7.	Epistemic Alignment: A Mediating Framework for User-LLM Knowledge Delivery	arXiv	Clark et al., 2025	Conceptual framework analyzing user-specific prompt preferences and epistemic dimensions
8.	Optimizing Large Language Models: A Deep Dive into Effective Prompt Engineering Techniques	Applied Sciences	Son et al., 2025	Experimental comparison of prompt techniques across datasets using multiple evaluation metrics.

9.	Prompt Engineering Guidelines for LLMs in Requirements Engineering	Journal	Ronanki et al., 2025	Systematic literature review combined with expert interviews for domain-specific guideline extraction
10.	Exploring the Potential of Prompt Engineering: A Comprehensive Analysis of Interacting with Large Language Models	ICCUBEA Conference	Pawar et al., 2024	Comparative analysis of ChatGPT and Bard using structured prompts and performance evaluation

III. SYSTEM ARCHITECTURE

A. Overall Architecture Overview

Starting at ground level, PromptX stacks functions like building pieces, each focused on a single job - reviewing inputs, organizing entries, or offering fresh ideas - all within a shared environment. At the front end, interaction begins visibly, guiding contributions inward toward areas where choices form gradually. Beneath initial steps, hidden sections apply learning systems to evaluate content while tuning into feedback people give after testing outputs. Worth is not assumed; instead it surfaces through combined automated review and accumulated user responses. Information travels in loops, adjusting scores, sharpening phrasing, advancing performance slowly but steadily. Out of sight, pieces fit together closely, keeping things fast without losing precision. One step leads to the next, clean and distinct, still able to shift if circumstances change. The result? Ideas formed through real use, multiple tests, changing patterns over time. Not one part does everything - power spreads evenly between layers. Tough decisions break down into simpler moves, revealing results that make sense, easier to follow.

From the highest layer down, there are five core parts linked in sequence: Presentation leads, while Application follows close behind. Instead of stopping there, it flows into Evaluation where checks happen. Further on, Data takes shape beneath that stage. Below everything sits Ranking and Recommendation forming the base. When a request begins, entering through Presentation, it shifts into Application right away. Only after that does it move toward Evaluation for closer look. Depending on what's found during analysis, sorting occurs - guided partly by automated scores, partly by how people interacted before. Personalized results appear since ranking combines with traces of earlier behavior. This kind of setup bends without breaking, expands quietly, handles change without needing constant adjustment.

B. Presentation Layer (User Interface Layer)

Right where you see things happen, that's the spot this layer takes charge. From here, people send in their ideas, look through others already shared, then get pointed toward useful picks. Searching shows up when needed, filters narrow choices down, ratings mark value, reviews share thoughts - each part connects smoothly. Everything typed gets bundled with background notes like past habits and personal likes before moving forward. Built quiet but smart, it keeps steps light so reaching deeper parts of the system feels natural. Flow stays clean because clutter slows everything behind the scenes.

C. Application Layer

Right where users meet the system, the Application Layer takes charge, linking what you see to what runs behind. Instead it passes login checks, sorts incoming requests, then pushes messages through APIs. Once someone sends a prompt, off goes the signal - directed precisely to tools like the processor and evaluator. Without skipping steps, it keeps information aligned while talking to storage. Through quiet oversight, pieces across the setup stay in sync. Only here does everything connect without confusion.

D. Evaluation Layer

What makes the PromptX system work begins with its Evaluation Layer. This part checks how good a prompt really is. Inside it lives several pieces that do different jobs. One piece handles incoming prompts

right away. Another uses artificial intelligence to score each one by looking at things like clear meaning, topic fit, and real-world impact. A third examines what users actually do when they interact. From those actions, it pulls numbers tied to performance - how much faster tasks finish, fewer tries needed, or often prompts lead to useful results. Each module connects but works on its own rhythm. Out there, these numbers show how well prompts actually work. Once gathered, the evaluations move on - funneled into the ranking setup for what comes next.

E. Data Layer

The Everything given by users gets saved through specific parts of the system. Stored information stays structured across multiple layers for ongoing availability. As usage grows, speed matters; separate databases hold distinct types: one tracks what people enter, another logs assessment outcomes, while a third preserves every exchange between person and platform - these exchanges subtly shape later behavior. Access spans broad ranges of engagement depth, making sure details do not disappear. Indexing methods arrange vast inputs logically, enabling rapid location despite volume. Updates occur instantly when new contact happens, adjusting results based on live input. Retrieval remains efficient because organization adapts without delay. Each entry persists in ways aligned with its type and purpose. Structure supports both immediate needs and long-term patterns equally well. No piece vanishes after being entered into active memory spaces.

F. Ranking and Recommendation Layer

Quality prompts reach users through careful sorting within the Ranking and Recommendation Layer. A mix of AI judgments, how people respond, plus real-world results shapes each score given by the engine. As usage shifts, so do positions - rankings shift with live trends instead of staying fixed. Personalized picks emerge when past choices meet current standings behind the scenes. Preferences guide adjustments quietly, making future suggestions feel more fitting without announcement or fanfare.

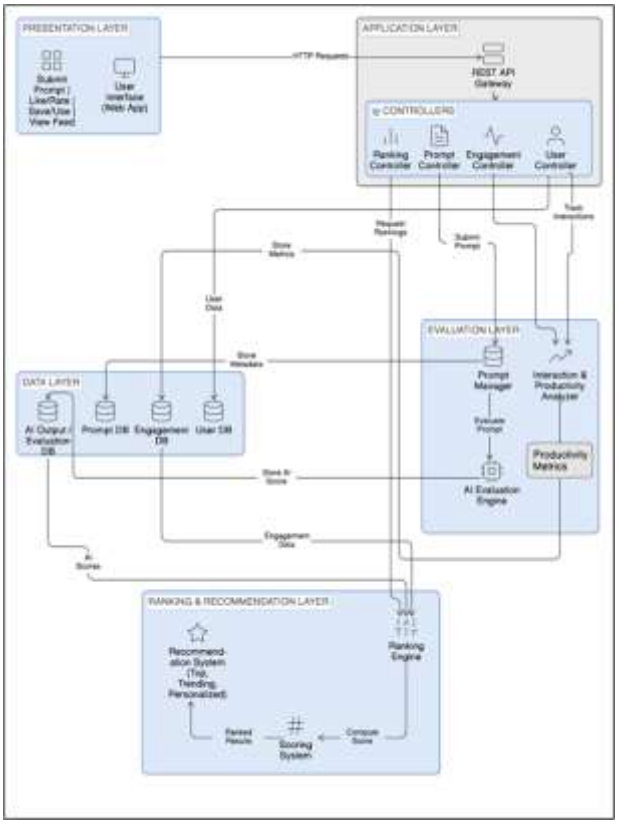

Fig. 1 System Architecture

IV. METHODOLOGY

A. Prompt Processing And Evaluation

Starting at the user's end, someone types a request into the interface. Once entered, the system hands it off to the Prompt Processing Module instead of moving forward unchecked. This step shapes the input, checks its format, then gets it ready - no skipping ahead. From there, the AI Evaluation Engine takes over, studying the rewritten version closely. Clarity matters here, also how fitting and strong the prompt feels when tested. What comes out is tied directly to how useful the result might be for a large model working hard behind scenes. A number appears after all this, standing quietly as proof of quality. Right from the start, every prompt gets scored using the same clear rules so results stay fair. Because of this setup, each result lands in the evaluation database without exceptions. One key detail stands out - rankings depend heavily on that number pulled straight from storage.

B. User Interaction and Productivity Analysis

Beyond artificial intelligence scoring, the tool watches how people actually use prompts day to day. Behavior gets logged - how often a prompt is picked, scored high, favored, or ignored over time. From that flow of actions, signs emerge: fewer tries needed to get it right, quicker results, more consistent wins. What sticks works better; what fails fades quietly. Clear signals show which inputs help users move faster through their tasks. Over time, watching how users engage can reveal which prompts work best. Stored in the database, that engagement info flows together with AI ratings to shape what happens next. Metrics pulled from actions blend quietly into the system's choices.

C. Ranking Mechanism

What shows up first depends on a mix of machine ratings and how people actually use the prompts. Instead of relying on one signal, the system blends several measurements - like engagement and efficiency - to form an overall mark. A stronger score means it tends to work better, so those appear earlier. Effectiveness isn't just about design - it also includes how well it performs in practice. Every time fresh interactions come in, the rankings shift on their own. Because of that, suggestions stay sharp when habits change. What you see today might look different tomorrow - patterns evolve, so do the results.

D. Recommendation System

From habits you show, it picks up what might interest you next. Patterns in your choices guide its next moves quietly. Out of old clicks and scrolls, ideas take shape slowly. Top ones rise without noise, simply by being chosen often. What others engage with flows into your view when fit is clear. Suggestions shaped just for you appear between the lines. Searching less becomes natural after a while. Guessing fades as better options keep arriving instead. Over time, it picks up habits from how people interact. Because of this learning curve, what gets suggested stays close to what each person actually wants. As a result, using the system feels smoother and makes more sense day after day.

E. Data Management and Storage

From every step come records - prompts, user details, ratings, how people engage, performance numbers - these settle into the data layer. Built neatly inside organized systems, storage stays swift, access smooth, growth possible. Inside one spot called Prompt Database live both what was asked and its background notes. Ratings made by AI, along with output measures, find home in another space named Evaluation Database. When users click, respond, or move through tasks, those traces fill the Engagement Database. A separate vault, the User Database, holds personal setups and individual choices. When handling massive amounts of data, speed comes from smart indexing paired with fast query tools. Performance stays steady because the way information is organized backs up instant analysis without delay.

V. IMPLEMENTATION DETAILS

A. Technology Stack

Running on a standard client-server setup, the whole PromptX platform works as one connected web experience, splitting tasks between what users see and how inputs get processed behind the scenes. Built with Node.js under the hood, the back end keeps things moving smoothly when dealing with live requests, while also running specialized methods that sort and score each prompt. User entries, feedback ratings, activity records, and personal details are all kept in MongoDB, chosen for its speed when pulling up info thanks to smart indexing. Instead of working directly with raw data shapes, the app relies on Mongoose to define clear templates, helping keep information organized and searches quick.

Built on Next.js, the frontend loads pages fast by generating them on the server before they reach users. Instead of one big chunk, it splits tasks into separate parts - each handles prompts, rankings, or how people interact. Data moves between front and back through structured API calls that keep everything in sync without delays. Speed, growth room, and long-term upkeep are baked in thanks to the tools chosen for building it. Each piece works

together quietly so nothing feels slow or clunky during use.

B. Prompt Evaluation and Ranking Integration

A fresh look at each prompt begins when someone submits one - intelligence built into the engine checks how clear, relevant, and strong it is. Scoring happens automatically, guided by real-time analysis instead of guesswork. After that, numbers from actual use step in: how often people interact, what they rate, where attention lingers. Quality isn't decided just once; repeated behavior shapes the outcome over time.

A single score decides each prompt's place, shaped by machine learning outputs along with actual usage signals. Instead of relying solely on abstract ratings, live feedback helps shape how prompts rank. Stored inside the database, these outcomes shift when fresh activity arrives. With every update, the process sharpens what works while keeping rankings current.

C. Data Storage and System Optimization

Handling lots of prompt and interaction records begins with smart organization behind the scenes. Stored inside MongoDB, prompts live alongside ratings, user details, and activity logs - structured but adaptable. Speedy access happens thanks to indexed entries that sort prompts by score without delay. Mongoose steps in to tighten queries and check data shape, quietly lifting efficiency across operations. Performance gains emerge simply from how pieces fit together, not extra power or complexity.

Handling many users at once works well because Node.js deals with tasks separately, without waiting. Because of this setup, the system keeps running smoothly even when loads increase unexpectedly. API keys stay protected by storing them outside code, using special settings known only to the server. Security improves since sensitive details never appear directly in files visible to everyone. Updates to rankings happen instantly, thanks to how quickly data moves through organized parts. Each piece of the design fits together but can change without breaking anything else. Performance remains steady under different conditions, not just ideal ones. Changes later on become easier since pieces work independently yet connect when needed. Real-time feedback stays accurate because information flows without delays or bottlenecks. The whole structure adapts quietly, without demanding attention or complex rework.

VI. EXPERIMENTAL SETUP AND EVALUATION

To evaluate the effectiveness of the proposed PromptX system, a series of controlled experiments and user-interaction-based evaluations were conducted. The evaluation focused on prompt quality assessment, ranking effectiveness, recommendation accuracy, and the overall impact of integrating AI-based evaluation with user interaction and productivity metrics. The experiments were performed using a curated dataset of prompts collected from various domains such as content generation, coding, and general problem-solving tasks. These prompts were stored in the system database and evaluated through the PromptX pipeline. Test scenarios were designed to simulate real-world usage, including prompt discovery, prompt reuse, and iterative prompt refinement to achieve desired outputs.

A. Evaluation Metrics

To test how well the new system worked, researchers looked at three main things - a rating for prompt quality, gains in productivity, and how fast the system responded. Instead of just one method, they used several factors like clear wording, usefulness, and impact strength to judge each prompt through automated scoring. Rather than guessing outcomes, actual user behavior shaped the findings when checking progress in work speed. Fewer repeated attempts, quicker paths to correct answers, and more frequent successful uses helped show real changes in daily tasks. Data pulled straight from live interactions made sure conclusions matched what happened, not assumptions about it.

What mattered most? How fast the system reacted. Time spent handling requests, sorting results, then delivering suggestions got checked closely. Behind the scenes, delays in data fetching, server tasks, and

external calls were tracked piece by piece. Taken together, those numbers showed how well the setup ran - while hinting at real gains in daily workflow speed.

B. Baseline Comparisons (Basic System vs AI Evaluation vs Full PromptX)

To analyze the contribution of different system components, three configurations were evaluated:

1. Basic Prompt System: Prompts are used without evaluation, ranking, or recommendation.
2. AI Evaluation System: Prompts are evaluated and scored using the AI Evaluation Engine.
3. PromptX (Proposed System): Full system with AI evaluation, user interaction analysis, ranking, and recommendation.

Sometimes a first try fails when nothing checks if it works right. Scoring steps helped sort good attempts from weak ones later on. Mixing scores with how people used them plus their output made one version stand out clearly. Even though that mix took more time to run, doing things that way paid off well enough to make the wait worth it.

C. Case Study: Prompt Optimization Workflow

A hands-on example took shape through a trial meant to mirror how people actually use prompts. Some participants got batches of prompts over several rounds, tackling things like creating written material, producing code, or working through questions step by step. With each round, they shaped their inputs again and again - tweaking them until results improved.

A closer look at how people used PromptX showed that suggestions got better the more they interacted with it. Results came faster because each try built on earlier ones, slowly cutting down wasted attempts. Suggestions popped up based not only on scores from algorithms but also on what actual users did before. As habits shaped the feed, finding useful prompts felt less like searching and more like recognizing familiar paths. Progress wasn't forced - it grew out of repeated use and quiet adjustments behind the scenes.

D. Performance Analysis

It turns out the test outcomes show something clear. When AI scoring joins forces with how users actually interact, prompts start working better. System behavior shifts in a positive direction at the same time. Instead of treating all inputs the same, the mixed-ranking method filters out weak attempts quietly. Stronger ones rise without extra nudging. Work speed gains become visible after a while. Guessing drops off sharply. Finding what works takes less time now.

Timing checks revealed quick response times during prompt sorting and scoring, thanks to smooth server performance alongside well-tuned data lookups. Suggestions felt natural, adapting to individual choices without delay. In total, PromptX struck a practical middle ground - handling output quality, speed, and ease of use together, growing steadily while staying dependable for handling prompts.

VII. RESULTS AND DISCUSSION

A. Results

Out of every test run, the new PromptX setup pulled ahead when it came to sharper prompts and faster work rhythms. Instead of guessing, people leaned on clear feedback from the built-in scoring tool. That score? It judged each prompt through how sharp it hit the topic, whether it fit the task, plus if it sparked useful replies. When prompts earned top marks, answers followed suit - closer to what users actually needed. Higher numbers meant fewer mistakes, less guesswork downstream.

Because it mixed AI scores with how people actually used things, the ranking method worked better. Real use mattered just as much as design ideas when sorting prompts. Top choices stayed on top, thanks to feedback from actual work patterns. What got picked most was what helped users get tasks done smoothly.

One thing stood out - people needed fewer tries to get the output they wanted. It didn't take as long to figure out which prompts worked well. What happened next? They got good results faster than

before. Fewer shots, more hits, simply because the right prompts came quicker.
Because of smart suggestions, finding good prompts took less time. When the system learned what users liked, it became easier to use. What happened shows PromptX makes prompts work better while asking less from people.

B. Discussion

What stands out is how well AI assessment works when mixed with actual user behavior and output tracking. Instead of just judging prompts on paper, the method ties together different aspects to show what really happens during use. By doing this, it checks if prompts do more than look good - they need to perform too.

One less stumble through guesswork shows how sorting prompts by structure adds value. Because guidance steers people to stronger inputs, wasted steps drop while progress gains momentum. When numbers tracking output enter the picture, they offer insight into performance that goes past mere correctness.

Even with extra steps for checking and sorting, delays stay low because the backend works fast and databases run smoothly. Performance holds steady, yet outcomes get better over time. Though more stages are involved, speed does not suffer thanks to smart handling behind the scenes.

A closer look at PromptX shows it handles speed, precision, and user experience without favoring one too much. Because it combines assessment, sorting, and suggestions, managing prompts feels complete. Results back up the method used while pointing out how it might boost output in apps powered by artificial intelligence.

VIII. FUTURE WORK

Though the suggested PromptX setup already supports smart prompt analysis and suggestions, extra upgrades could appear down the line. Moving beyond basic checks, deeper AI reviewers might grasp nuances like purpose or subject area much clearer. Because of this shift, scores given to prompts may reflect real usefulness more closely. At the same time, handling inputs across formats - like pictures alongside words or programming snippets - could become possible too.

One key focus ahead involves improving customization using smarter models of user behavior. With machine learning, adaptation to each person's tastes becomes sharper, leading to tighter suggestions. On top of that, live data tracking along with instant feedback loops allows the setup to evolve constantly, refining how prompts are ranked over time.

One way forward could be turning the space into a place where people trade prompts, adding tools to earn from strong contributions. Built-in ratings might help highlight what works while opening doors to broader sharing. When systems grow smarter in handling load, more users arrive without slowing things down. Real-time responses gain speed when architecture shifts happen behind the scenes. Strength builds slowly - each upgrade nudging output quality higher across daily AI tasks. Efficiency finds room when infrastructure stretches without breaking rhythm.

IX. CONCLUSION

This study introduced PromptX - a tool powered by artificial intelligence that helps builders and developers find, test, and apply effective prompts. Instead of relying on basic sharing features like older systems do, it weaves together smart assessment tools, how people engage with content, and real-world performance data under one roof. With a mix of adaptive scoring logic paired alongside tailored suggestions, finding strong prompts becomes quicker, sharper. What stands out is how each piece feeds into the next, quietly boosting results without drawing attention.

It turns out the tests show better prompts come through less guessing, thanks to fewer tries needed up front. What stands out is how quickly useful prompts appear now, which users notice right away during regular tasks. One thing leads to another when success happens more often - each round

works smoother than before. Efficiency sticks around, no matter how much info flows in or how many people jump in at once. Performance holds steady because speed stays high without skipping steps.

A fresh look at how prompts are handled comes through in PromptX, shaping things with clarity and purpose. Its smart design quietly lifts the way users interact with AI tools. Not loud, just effective - growth happens without shouting about it. Work flows smoother when systems adapt instead of resist. Across fields like research or creative tasks, the framework stretches where needed. Strength hides in its quiet flexibility, showing up most when complexity rises.

X. ACKNOWLEDGEMENTS

The authors would like to express their sincere gratitude to the project supervisor and faculty members of the department for their continuous guidance, constructive feedback, and valuable suggestions throughout the development of this work. Their insights greatly contributed to refining the system design, methodology, and overall research direction. We also acknowledge the support provided by our institution for offering the necessary infrastructure, computational resources, and academic environment required to carry out this project successfully. Special thanks are extended to peers and colleagues who provided helpful discussions and testing support during the implementation and evaluation phases. Finally, we recognize the contributions of the open-source community and the developers of the frameworks and tools utilized in this work, which made the development and implementation of the PromptX system possible.

REFERENCES

1. S. Bhatt et al., "From Prompts to Performance: Leveraging LLMs for Enhanced Educational AI Interactions," in IEEE, 2025.
2. B. Zhou, X. Wang, and J. Liu, "Large language models are human-level prompt engineers (Automatic Prompt Engineer, APE)," arXiv preprint arXiv:2211.01910, 2022.
3. J. Liu, S. Wang, and Y. Zhang, "Prompting frameworks for large language models: A survey," arXiv preprint arXiv:2311.12785, 2023.
4. W.-L. Chiang, Z. Zheng, X. Sheng, and I. Stoica, "Chatbot arena: An open platform for evaluating LLMs by human preference," arXiv preprint arXiv:2403.04132, 2024.
5. M. Jacobsen and R. Weber, "The promises and pitfalls of LLMs as feedback providers," Journal, 2025.
6. C. Chen, Z. Li, and H. Xu, "Unleashing the potential of prompt engineering for large language models," arXiv preprint arXiv:2310.14735, 2023.
7. K. Clark, J. Smith, and L. Brown, "Epistemic alignment: A mediating framework for user-LLM knowledge delivery," arXiv preprint arXiv:2504.01205, 2025.
8. J. Son, Y. Kim, and H. Park, "Optimizing large language models: A deep dive into effective prompt engineering techniques," Applied Sciences, vol. 15, no. 3, 2025.
9. K. Ronanki, A. Sharma, and P. Gupta, "Prompt engineering guidelines for LLMs in requirements engineering," Journal, 2025.
10. V. Pawar, S. Kulkarni, and R. Patil, "Exploring the potential of prompt engineering: A comprehensive analysis of interacting with large language models," in Proc. ICCUBEA, 2024.