Transactions on Computer Science and Intelligent Systems Research
ISSN: 2960-1800, eISSN: 2960-2238 | Volume 9
AIDML 2025

Design and Application Research of Dynamic NPC System Based on Local Large Language Models

Xinrui Li *

College of Computer Science, Beijing Technology and Business University, Beijing, China

* Corresponding Author Email: 2207070215@stu.btbu.edu.cn

Abstract. With the Development of the video game industry, the problem of traditional Non-Player Characters (NPCs) became rigid because they rely on pre-written scripts. Although the Large Language Models (LLMs) offer a new way to generate dynamic conversations, most useful ways nowadays from using cloud-base solutions suffer from high response delay (>500ms) and privacy risks. Recent research has highlighted the need for on-device AI solutions to address these performance and security concerns. This study explores a local deployment approach by using the DeepSeek model with localized model compression, the paper successfully ran LLM-powered NPC dialogues within the game environment. Tests show that our method reduces response delay to 14% of the cloud-based methods. Meanwhile, it achieves diversity scores of 0.22 and 0.35 based on TF-IDF and BERT, significantly enhancing the use of using LLMs in video games. Our findings demonstrate that this approach significantly improves the use of LLMs in games. It provides a low-latency, privacy-friendly solution and presents a new way to integrate AI into the gaming industry.

Keywords: Large Language Model; NPC; Viedo game; Artificial Intelligence.

1. Introduction

With the rapid development in the Video Game industry, the expectations for immersion and freedom from players also have significantly increased. However, because of script interactions, traditional non-player characters always appear rigid in interactions with players, cannot to meet player's demands for dynamic interactions. But in recent years, breakthroughs in large language models (LLMs) have opened up the new possibilities for addressing this issue.

As the groundbreaking advancements in natural language processing technology, LLMs have gradually transitioned from theoretical to practical application facing the general public. Since OpenAI introduced GPT-1 in 2018 [1], the parameter scale has grown from 117 million to 671 billion in DeepSeek-R1, leading to improvements in text processing and generation quality.

Against this backdrop, some studies have begun exploring connecting LLMs to the video game industry to use of GPT-driven NPCs to enhance player experiences [2-4]. For example, one study proposed a modular framework based on ChatGPT models, by bringing in LLMs to enhance NPC autonomy and thereby improve player immersion [5]. Additionally, some research has focused on improving the contextual relevance of NPC dialogues and their interactions with players based on predefined backgrounds, as well as increasing player engagement [6]. Besides, one study experiment demonstrated the potential of context-aware dialogues in enhancing the player experience [7]. Another comparative study examined voice recognition versus traditional dialogue boxes as interaction methods, showing the differences in immersive experience [8]. However, some studies also highlighted the negative impact of response latency on overall gameplay experience [9].

However, in the past, the quality of dialogue generated by LLMs was often in direct proportion to the number of parameters. Due to heavy demands on memory and GPU usage for read/write operations, they could not be deployed on standard home computers, and were limited to cloud-based deployment models with high latency and privacy risks, which make them hard to use in real-time interactive scenarios.

In 2025, the DeepSeek team released the DeepSeek-R1 model, which innovatively introduced a reward mechanism into the training process of LLMs [10]. By using knowledge techniques to make it possible to train LLMs with a smaller number of parameters and maintain the high dialogue quality. After distilling the model using Qwen, the parameter count was reduced to the 7-billion level, which making it possible to deploy LLMs locally on consumer-grades computers.

This study is based on that technology foundation, and targets core challenges in the video game industry, such as rigid NPC dialogues and a lack of immersion. This study developed a locally-deployed LLM-driven dynamic interaction system and validated its technical feasibility and practical value.

2. Research Design and Methodology

2.1. Core research Objectives

2.1.1. Dynamic Dialogue Generation

Traditional NPCs' interactions in games are always limited on scripted responses. By using locally deployed large language models to power NPCs, it can enable NPCs to have dynamic conversations with players.

2.1.2. Performance Optimization

By utilizing the DeepSeek model and other emerging technologies, this study makes transitions LLM inference from cloud-based APIs to local execution, it ensures basic conversation quality while eliminating delays and errors caused by network instability or server fluctuations. Additionally, it prevents security risks such as data leakage.

2.2. Technical Approach

As shown in Figure 1, this study is divided into four phases: Model Deployment Phase, Game Development Stage, Integration and Testing Phase, and Result Analysis Phase. In these four phases, this study will deploy a locally deployed large language model, developing a simple 2D game demo featuring NPC dialogues powered by local LLM, and survey comparative experiments. The locally deployed LLM-driven NPC dialogue system will be compared with traditional cloud-based models, other local models without using DeepSeek to distillation, and manually scripted dialogues. Finally, I will analyze the results and discuss future directions.

3. System Implementation and Key Technologies

3.1. Model Selection and Deployment

This study uses the Ollama framework to locally deploy the DeepSeek-R1 (7B) model. Because of Ollama supports macOS, Windows, and Linux, making it a mature solution for local LLM deployment while minimizing resource consumption. After deployment, the paper tested about system resource usage. When running DeepSeek-R1 (7B) on an NVIDIA RTX 3060 Laptop, the peak VRAM consumption was only 4.2GB, meeting the lightweight requirements of this study.

Additionally, Ollama provides a fully functional local API with output accessible via HTTP POST, which follows the same format as OpenAI API, which can ensure high compatibility and simplifies integration between the LLM and the Unity Demo in the experiment.

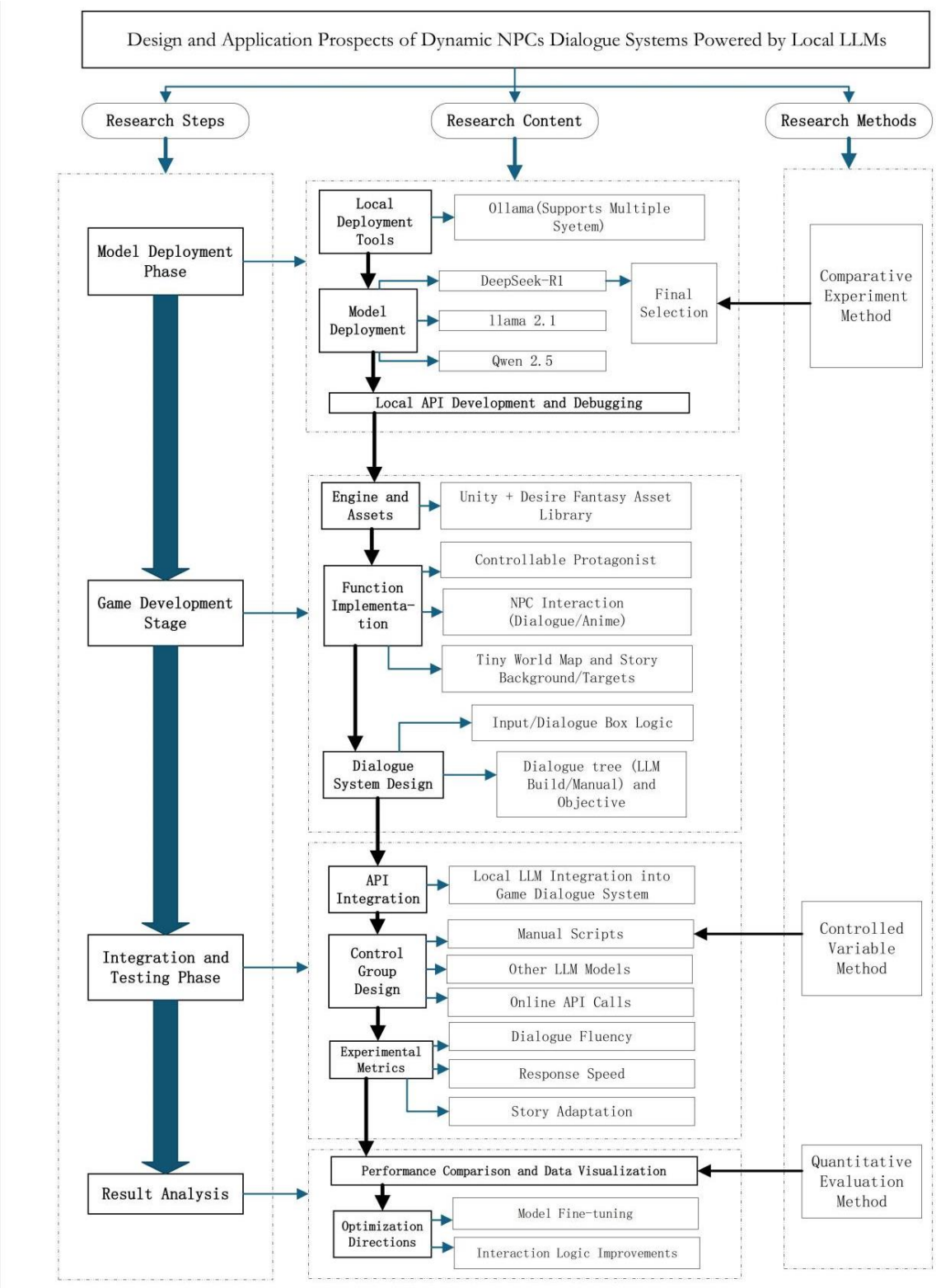

Figure 1. Technical Approach (Picture credit: Original).

3.2. Game Development

3.2.1. Engine and Assets

This study demo is based on Unity 2022.3.20f1 (LTS Version). The 2D scenes and character models are sourced from the open-source asset pack *Desire Fantasy* on *itch.io*. The game is programmed based on C#, which coding about separate modules for character interactions and core dialogue and LLM management. This lightweight game demo was created to evaluate the usability of the LLM model.

3.2.2. Dynamic Dialogue System Design

In the demo, we set the game targets for the player, player can freely input text through a dialogue box to interact with NPCs to achieve the targets, rather than selecting from predefined choices. To ensure that the generated dialogues align with expectations, background constraints were applied to the LLM in Unity by embedding NPC-specific background settings.

3.3. System Integration

Ollama's built-in local API is used to facilitate real-time LLM interactions in the games. By using the OpenAI-compatible API provided by Ollama and directly embedded into the script, it allows NPCs to dynamically respond to player inputs by invoking the locally deployed language model.

4. Experiment Design

4.1. Experiment Setup

4.1.1. Comparison Methods

Table 1. Comparison of Testing Schemes.

Group	Model/Method	Deployment Method
Experimental Group	DeepSeek-R1:7B	Local (Ollama)
Control Group 1	DeepSeek-R1:671B	SiliconFlow API
Control Group 2	Llama 3.1:8B	Local (Ollama)
Control Group 3	Qwen 2.5:7B	Local (Ollama)
Control Group 4	Manual Script	Built-in Unity

As shown in Table 1, this study selects four common methods for driving NPCs in the game: the full DeepSeek-R1 cloud API provided by SiliconFlow.cn, local light models Llama 3.1: 8B and Qwen 2.5: 7B, and traditional manually scripted NPCs. These methods are compared with the locally deployed DeepSeek-R1 model through a series of experiments.

4.1.2. Testing Scenarios

To objectively evaluate the differences in response latency and error rates between locally deployed DeepSeek (experimental group) and cloud-based API (control group 1), three typical dialogue tasks were be designed:

1. Simple Queries (e.g., "Hello Mr.Dog!").

2. Background-related Queries (e.g., "Do you know about Mipha?").

3. Complex reasoning (e.g., "Where should I go to find Mipha?").

These dialogue tasks were automated using Python scripts, which can simulate player inputs to interact with NPCs embedded with predefined backgrounds. Response diversity was being recorded, and each test was repeated 50 times under stress testing conditions to measure error rates.

Additionally, to compare local DeepSeek-R1 (experimental group) with other lightweight local models (experimental groups 2 & 3) in terms of response diversity and consistency, like response latency test, this study also sets three rounds of NPCs background-related dialogues. The responses were compared with manually written scripts using TF-IDF and BERT-based methods to measure diversity and keyword coverage.

4.2. Performance Comparison Results

4.2.1. Compared to Cloud-based API

With the test result, the study found that compared with Cloud-based API, locally-based DeepSeek-R1's stable service is ensured with no 402 errors, insufficient API credit issues, or server overload problems, while response time is significantly reduced, achieving near-zero latency.

Table 2. Economic Data Statistics.

Input	Local Latency	Online Latency	Error (Local)	Error (Online)
Hello Mr.Dog!	7419.06ms	173.01ms	None	HTTP 503 (Once)
Do you know about Mipha?	5581.77ms	5752.93ms	None	None
Where should I go to find Mipha?	3524.11ms	23798.00ms	None	None

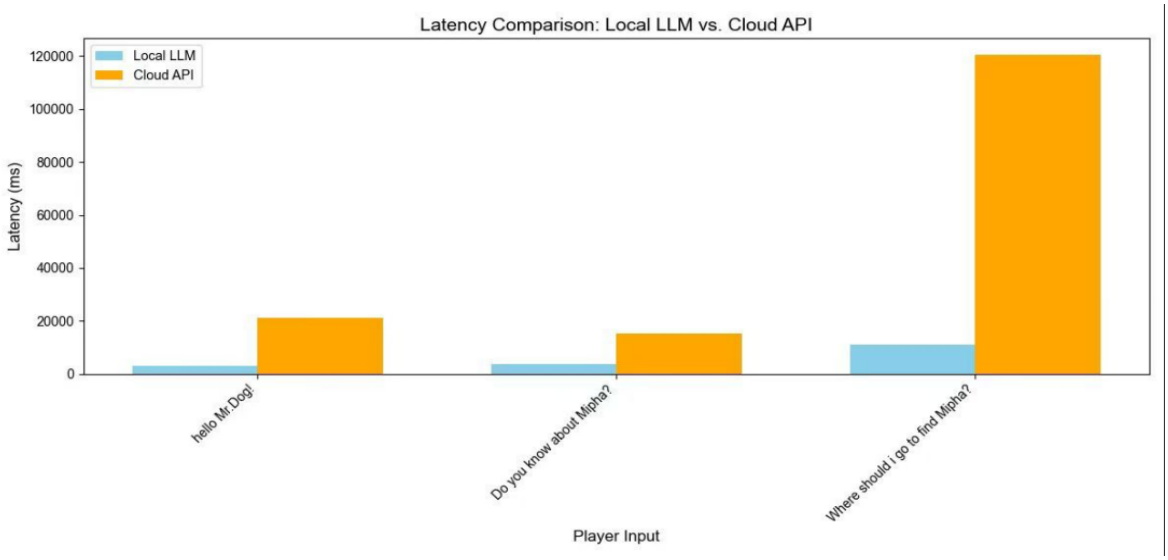

Figure 2. Latency comparison: local vs. online LLM (Picture credit: Original).

Table 2 presents the latency test results, and Figure 2 illustrates the comparison in bar charts. In Figure 2, blue bars represent local API calls, while orange bars represent cloud-based API calls. These Table and figures shows that local API latency is consistently lower than cloud-based API latency. In background-related queries and complex reasoning queries, local LLM exhibits even greater latency advantages.

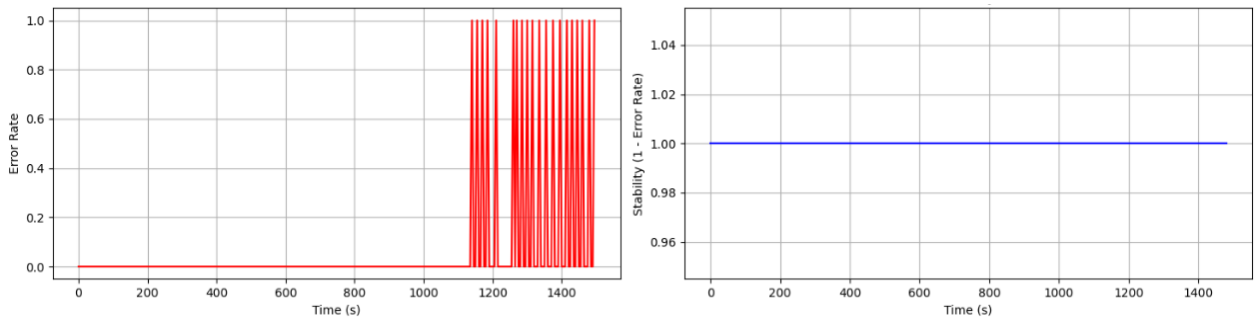

Figure 3. Cloud API Error Rate Over Time and Local LLM Stability Over Time (Picture credit: Original).

Additionally, Figure 3 depicts the change in error rates over time. In the graphs, the left graph represents the cloud API, while the right graph represents the local API (Y-axis: 1/error rate). With the time growing, cloud-based APIs encounter errors more frequently, whereas the local API remains error-free during stress testing.

4.2.2. Compared to Other Local LLM Models

In these tests, the study found that DeepSeek-R1 powered NPCs provide more human-like reasoning and responses and provide better execution of developer instructions for NPC replies than the other local LLMs powered NPCs.

Table 3. Comparison of Dialogue Diversity and Keyword Coverage Across Models.

Model	Diversity (TF-IDF)	Diversity (BERT)	Keyword Coverage	Average Similarity with Examples
Llama3.1:8B	0.08	0.30	66.7%	0.38
Qwen2.5:7B	0.15	0.25	72.2%	0.40
DeepSeek-R1:7B	0.22	0.35	77.8%	0.37

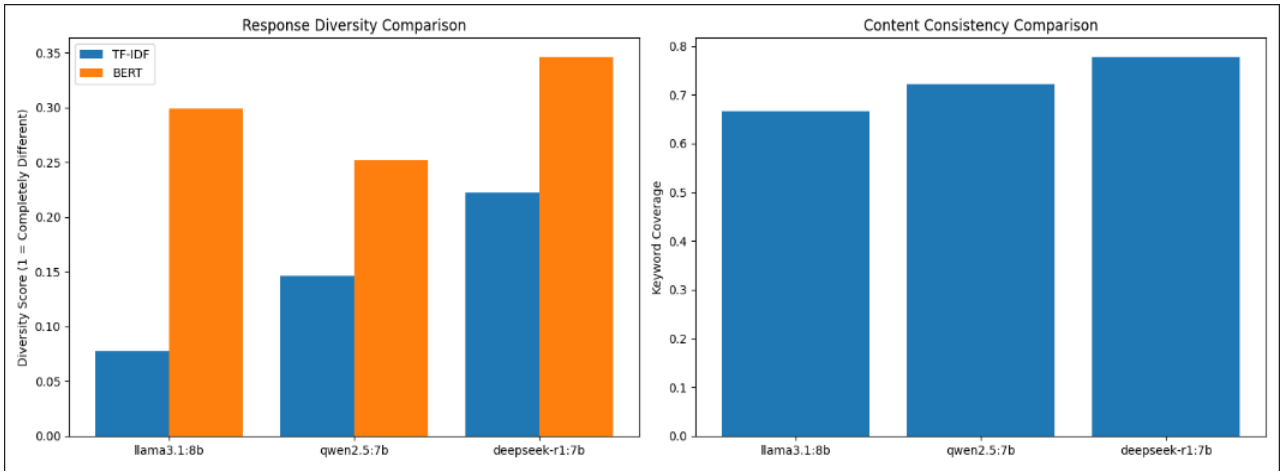

Figure 4. Response Diversity Comparison and Content Consistency Comparison (Picture credit: Original).

In those Table 3 and Figure 4, Diversity (TF-IDF) means measures variation in generated responses based on term frequency-inverse document frequency, which higher values indicate greater diversity; Diversity (BERT)means evaluates semantic diversity using BERT embeddings, higher values indicate richer content; Keyword Coverage means the proportion of developer-defined keywords appearing in generated responses; average Similarity with Examples can measures cosine similarity between generated responses and predefined samples, lower values indicate greater originality.

The results show that DeepSeek-R1 powered NPCs achieve the highest diversity scores, with 0.22 in TF-IDF and 0.35 in BERT-based evaluation. It also demonstrates superior keyword at 77.8%, better than Llama3.1 (66.7%) and Qwen2.5 (72.2%). These findings suggest that DeepSeek-R1's unique training approach enhances text coherence, improves keyword retention, and generates more diverse responses, making NPC interactions more immersive and suited for role-playing.

4.2.3. Compared to Other Local LLM Models

A user study (n=10) was conducted to compare LLM-generated NPC dialogues with pre-scripted dialogues. The results shows that participants generally preferred interactions with DeepSeek-powered NPCs, highlighting the greater flexibility and variety in conversations.

In terms of game target achieve rates, responses generated by the local DeepSeek model achieved a 90% success rate, almost the same as the pre-scripted dialogues and the cloud-based API model (control group 1). In the other hands, other locally deployed models performed slightly worse, with control group 2 achieving a 70% success rate and control group 3 reaching 60%.

5. Conclusion

This study addresses the limitations of traditional NPC interactions, including rigid dialogue responses because of the rigid script, high latency, and privacy concerns associated with cloud-based large language models. To overcome these challenges, the paper proposes a locally deployed solution using the DeepSeek R1 model to power NPC.

Experimental results show that the local deployment approach effectively reduces latency to just 14% of the cloud-based solution. Additionally, it outperforms other models powered NPCs in dialogue diversity (TF-IDF: 0.22, BERT: 0.35) and keyword coverage (77.8%). These advantages enable NPCs to provide more dynamic and responsive interactions, enhancing players freedom and immersion without compromising performance.

This research presents a viable paradigm for mixing large language models into gaming, offering a low-latency and privacy-saving solution. It also broadens the application scope of local LLMs in actual application scenarios. However, this study still has some limitations, such as the lack of advanced fine-tuning and model distillation. Future work can focus on personalized distillation for each NPC uses LLM models to further advance the adoption of generative AI and large language models in interactive entertainment.

References

[1] A. Radford, K. Narasimhan, T. Salimans, et al., Improving language understanding by generative pre-training, (2018).

[2] J. van Stegeren, J. Myśliwiec, Fine-tuning GPT-2 on annotated RPG quests for NPC dialogue generation, in: Proc. 16th Int. Conf. Foundations of Digital Games, (2021), pp. 1–8.

[3] M.F. Hasani, Y. Udjaja, Immersive experience with non-player characters dynamic dialogue, in: Proc. 1st Int. Conf. Computer Science and Artificial Intelligence (ICCSAI), IEEE, (2021), pp. 418–421.

[4] M. Deriu, F. Bachis, M. Massa, Improving the user engagement in a fully immersive experience by the means of a conversational non-playable character used as a tourist guide, in: 2021 IoT Vertical and Topical Summit for Tourism, IEEE, (2021), pp. 1–4.

[5] D. Ogunlesi, X. Wang, GPT-NPC: Enhancing NPC Human-Likeness and Autonomy in Video Games, (2024).

[6] B. Kim, M. Kim, D. Seo, et al., Leveraging Large Language Models for Active Merchant Non-player Characters, arXiv preprint arXiv: 2412.11189, (2024).

[7] L.M. Csepregi, The effect of context-aware LLM-based NPC conversations on player engagement in role-playing video games, Unpublished manuscript, (2021).

[8] F.R. Christiansen, L.N. Hollensberg, and N.B. Jensen, et al., Exploring presence in interactions with LLM-driven NPCs: A comparative study of speech recognition and dialogue options, in: Proc. 30th ACM Symp. Virtual Reality Software and Technology, (2024), pp. 1–11.

[9] M. Agarwal, A. Qureshi, L.L.N. Sardana, et al., LLM inference performance engineering: Best practices, URL: https://www.databricks.com/blog/llm-inference-performanceengineering-best-practices, (2023).

[10] D. Guo, D. Yang, H. Zhang, and et al., Deepseek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning, arXiv preprint arXiv: 2501.12948, (2025).