Developing a Multimodal LLM-Based Game Tutor for Hades

Yvette Lin
Stanford University
yvelin@stanford.edu

Raghav Ganesh
Stanford University
graghav@stanford.edu

Abstract

Roguelike games such as Hades often have difficult and challenging gameplay mechanics, hence, a game tutor helping players can improve player performance and reduce overall player frustration. In this project, we have developed an LLM-based game tutor for Hades that can be activated on demand while playing the game, to get real-time suggestions on how to modify gameplay strategy to maximize player success.

To serve this need, we use a combination of game data and multimodal image understanding to infer game state and build a RAG-based system with community-written strategy guides serving as our knowledge base. Our system is provides real-time gameplay advice grounded in a knowledge base comparable to that of an advanced player of the game, and capable of multi-turn conversation. We evaluate the system by through case studies of gameplay evaluated by an experienced player of the game, and find that our system is capable of giving useful gameplay recommendations grounded in the current game state and knowledge of gamneplay strategies.

1 Introduction

Video games generally have deep and varied strategies which players can use to achieve success. Learning these strategies is one of the biggest challenges players face during gameplay, and often where players experience the most satisfication and frustration. Generally, when faced with a level a player is simply unable to progress through, a common course of action is to go online and look for guides that might provide hints or guides as to how they can traverse and get past obstacles that face them. However, this process may be time-consuming, and it may be hard for a player to apply the advice in a static pre-written guide directly to the dynamic game state that they find themselves in.

Hence, our goal is to develop a conversational virtual agent which will have knowledge of the player and their game state to then provide *personalized* and accurate advice on how to progress through the game. Given the recent developments in large vision-language models (VLMs), our goal is to see if we can leverage these technologies to ensure that our agent returns grounded and meaningful responses.

To this end, we infer the current game state, leveraging a VLM to visually interpret the current screen and use a retrieval-augmented generation (RAG)-based system to generate advice and answer player queries. The RAG system is grounded in a knowledge base of online guides written by advanced players in the game's community.

We demonstrate our agent on the video game Hades, which is a popular "roguelike" game, whose gameplay consists of repeated (mostly disjoint) *runs* where the aim is to defeat a long gauntlet of enemies without dying. An For each run the player's *build* (consisting of their powers and abilities) is constructed throughout the run through a long series of choices from randomly-generated options throughout the run. We evaluate our agent through qualitative case studies, and find that it presents a promising framework for automated personalized advice to help players advance through a game.

2 Related Work

Existing work within the realm of the intersection of LLMs & Video Games (Gallotta et al., 2024), RAG (Lewis et al., 2021), and game tutors does exist, with prior research focusing on various aspects of this intersection. Several research papers we read focused on the entire to industry of video games and with relation with LLMs, but no one specific study focused a tutoring system similar to what we proposed.

With human game tutors, the expectation is that a skilled tutor would also be skilled at playing the game. As we are interested in developing a LLM driven game tutor, we are still interested in the possibility of having the LLM model itself be able to play the game well. Prior work in the field of AI for games such as with AlphaGo (Silver et al., 2016) and Agent57 (Badia et al., 2020) demonstrates that AI models are capable of playing simple games well, but when it comes to more complex games (in terms of both game mechanics and data modality), current AI models tend to perform worse. An example of this is Ireddy (2024), where a combination of a Yolo vision model and GPT-4 attempts to play God of War (2016). The model performs somewhat well, but performs a lot worse than the average human player, largely due to the complex visual stimulus on the screen and complex game mechanics. From these, we have decided to keep the AI agent being able to actually play the game outside of the scope of our CS 224V project, and solely focus on the coaching aspect of a tutor.

Cambrin et al. (2024) introduces an automated method of evaluating video game tutorial quality using Vision Langauge Models (VLMs). Due to the importance of effective tutorials for player engagement, especially in complex games, where the authors propose using VLMs to analyze tutorial frames and simulate human perception. The approach here involves extracting key frames from tutorials and posing questions a developer would ask a human game tester. The VLM responses are then compared with the expected answers to assess clarity and effectiveness.

do Nascimento Silva and Chaimowicz (2017) develops an AI-driven tutor for assisting players in Multiplayer Online Battle Arena (MOBA) games, specifically focusing on League of Legends. This AI tutor is an AI agent that also plays in the game as a supportive interactable AI NPC, that plays alongside human players, anlyzes their performance and also provides real-time tips to improve player performance. Through this system, the authors found that players were able to improve their understanding of game mechanics and strategies, and especially enhanced the experiences of newcomers.

3 Method

We develop a system capable of answering user queries regarding game tactics and strategies specific to the current game context. We infer the game state with a combination of stored internal game values relevant to the *run* and visual screen understanding via a large Vision-Language Model (VLM) and use a retrieval augmented generation (RAG)-based system with a knowledge base derived from community-written guides to return gameplay recommendations. Fig. 1 shows an overview of our method.

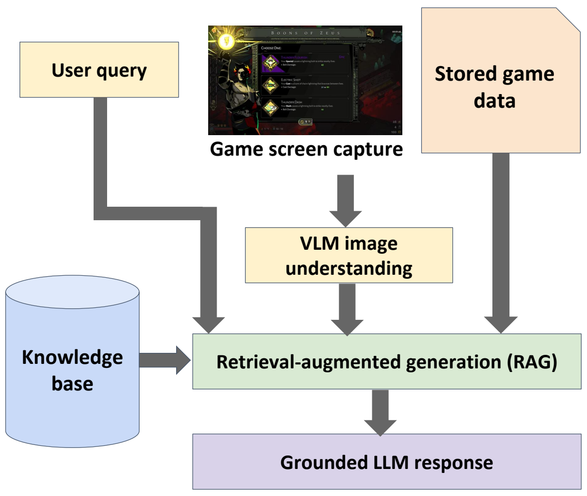
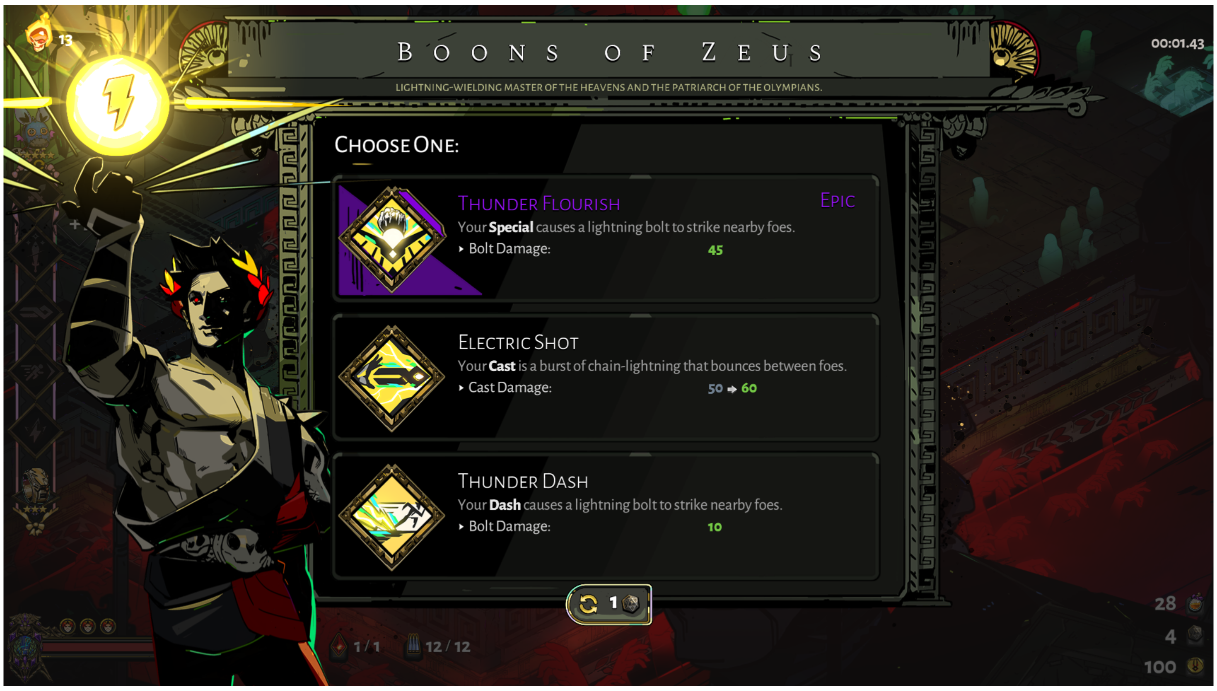


Figure 1: Method overview. We use a combination of game data and multimodal image understanding to infer game state and build a RAG-based system with community-written strategy guides serving as our knowledge base.

3.1 Understanding Game State

In Hades, in each *run*, the player starts in the Underworld and must fight their way through a long series of rooms to reach the surface of the earth. Most rooms consist of a combat encounter, where the player must defeat all the enemies to advance, at the end of which they are usually rewarded with an opportunity to increase the strength of their *build*, which consists of the player's current weapon, abilities, and powers. For example, after successfully completing a combat encounter, the player may be offered a choice of one of three abilities (called *boons*), and choose one that empowers their weapon's primary attack to do more damage. In this way, the player's build is slowly created throughout the run.

The current build is important to providing game recommendations personalized to the player's current *run*, and we extract it from by decoding the current auto-save file, yielding a list of strings B, with each $b_i \in B$ corresponding to an ability the player currently possesses.



Your **Special** causes a lightning bolt to strike nearby foes.
‣ Bolt Damage: 45

Your **Cast** is a burst of chain-lightning that bounces between foes.
‣ Cast Damage: 50 ➔ 60

Your **Dash** causes a lightning bolt to strike nearby foes.
‣ Bolt Damage: 10

Figure 2: An example choice of one of three *boons* as might be offered at the end of a combat encounter.

However, the player may ask a question that requires more immediate understanding—for example, when faced with a screen presenting the player with a choice of three possible *boons* (such as in Fig. 2), the player might wish to ask "What should I choose?" In this case, understanding the contents of the player's current screen is required. To this end, we take a screenshot of the current screen and query a VLM to summarize the contents. In practice, we observe that the vast majority of requests for advice involve asking a question about a set of choices, so for the purposes of this project, we query the VLM with the screenshot and the prompt "`What are the choices? Deliver your answer in Python list format using double quotes for strings.`" This yields a list of choices C.

3.2 Retrieval-Augmented Generation

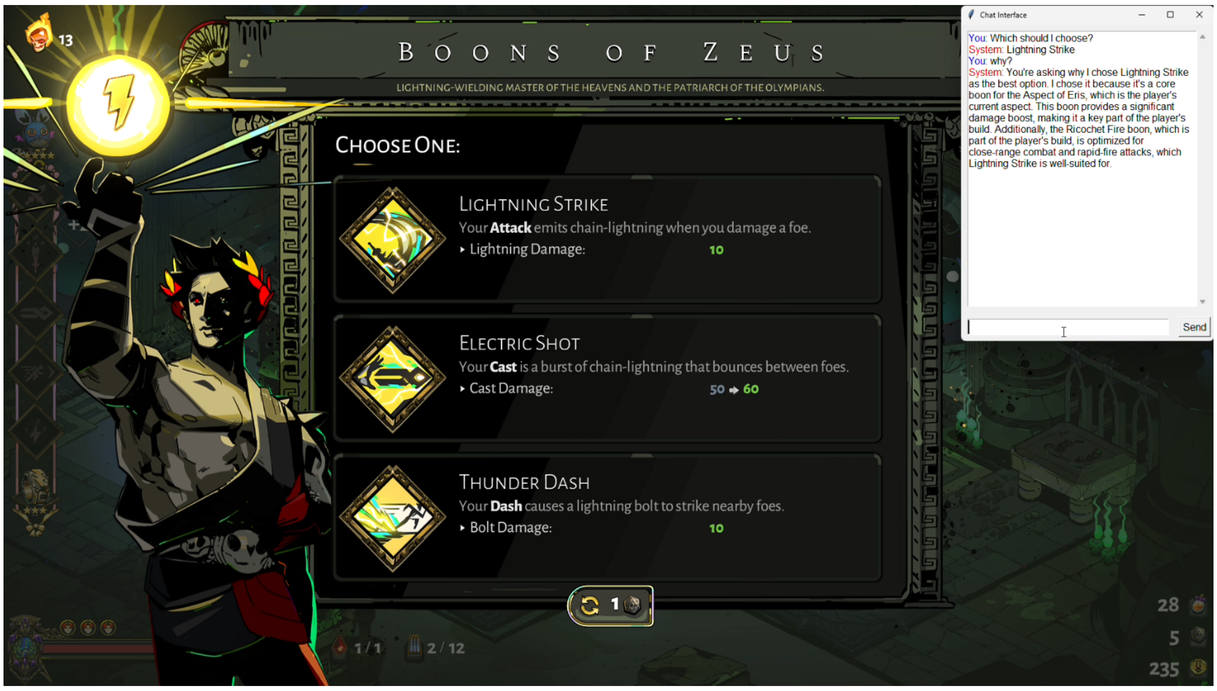
To build a knowledge base for the RAG system, we scrape the web for community-written guides on gameplay strategies for Hades. Notably, we draw from guides curated by the *r/HadesTheGame* forum on Reddit and the page for Hades on `speedrun.com`, as we expect these to provide reliable and "correct" advice.

To build the search system for RAG, we encode each of the document contents with a pre-trained text embedding model. Given a user query q, we concatenate the build B with q and search the knowledge base with Bq, yielding a context D consisting of the top k documents. Then, to generate the appropriate response to the user query, we use Chain-of-Thought reasoning as provided by DSPy (Khattab et al., 2024, 2022). We pass the documents D, the build B, the user query q, and the choices C to Chain-of-Thought and return the response a to the player.

3.3 Multi-Turn Conversation

The player may also desire to engage in additional conversation turns after the initial question, such as asking for further explanation for the advice recommended by the system. To this end, we additionally save the history $\mathcal{H}$ of the conversation between the player and the game tutor, crucially along with the context in which each message was sent. Concretely, throughout the conversation, for each player message q, we concatenate the build information B and the relevant choices C and add the string "`Me:` BqC" to $\mathcal{H}$. For each response a returned by the game tutor, we concatenate the Chain-of-Thought reasoning r which was used to generate a and add to $\mathcal{H}$ the string "`You:` a`[Your reasoning:` r`]`".

Then, given a player query q, to generate the next conversation turn (the response from the game tutor a), we pass the history $\mathcal{H}$ to the VLM along, and augment the query to read "`Me:` q`. You:` " to encourage the VLM to understand and integrate


BOONS OF ZEUS

LIGHTNING-WIELDING MASTER OF THE HEAVENS AND THE PATRIARCH OF THE OLYMPIANS.

CHOOSE ONE:

LIGHTNING STRIKE
Your **Attack** emits chain-lightning when you damage a foe.
‣ Lightning Damage: 10

ELECTRIC SHOT
Your **Cast** is a burst of chain-lightning that bounces between foes.
‣ Cast Damage: 50 → 60

THUNDER DASH
Your **Dash** causes a lightning bolt to strike nearby foes.
‣ Bolt Damage: 10

Chat Interface

You: Which should I choose?
System: Lightning Strike
You: why?
System: You're asking why I chose Lightning Strike as the best option. I chose it because it's a core boon for the Aspect of Eris, which is the player's current aspect. This boon provides a significant damage boost, making it a key part of the player's build. Additionally, the Ricochet Fire boon, which is part of the player's build, is optimized for close-range combat and rapid-fire attacks, which Lightning Strike is well-suited for.

Figure 3: An example interaction with our game tutor. Our system recommends the “correct" boon and grounds its explanation in knowledge about the current build.

the context-aware history $\mathcal{H}$.

4 Results & Discussion

4.1 Implementation Details

To create the search system described in Section 3.2, we use the embedding model `m2-bert-80M-8k-retrieval` provided by Together AI and search for the top $k = 2$ documents. For all VLM queries described in Section 3, we use `Llama-3.2-11B-Vision-Instruct-Turbo` provided by Together AI.

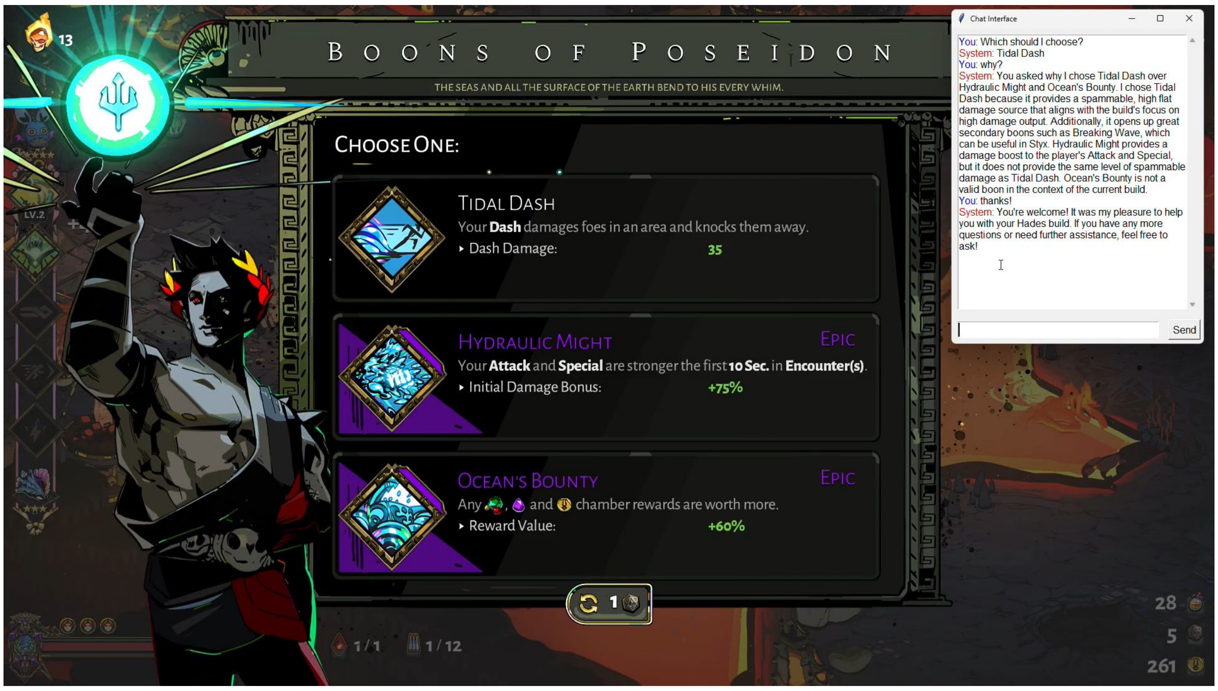
4.2 Evaluation

Evaluating our system quantitatively is difficult for several reasons. First, while there is certainly a body of commonly accepted effective choices and strategies for Hades among high-level players (hence, there are definitely “correct" and “incorrect" strategies), there currently exists no such dataset against which we can test the ability of our game tutor to answer user queries “correctly." Furthermore, user studies are difficult to conduct because the game Hades is not free-to-play (requires a purchase of about $25) and moreover only runs on sufficiently powerful Windows machines. Hence, it is difficult to solicit and fund volunteers for a large-scale user study.

Therefore, we evaluate our system qualitatively by examining case studies. (One of the authors considers themselves a fairly advanced player of the game[1] and hence feels qualified to evaluate game tutor recommendations from a fairly advanced perspective.)

In Figure 3 we show a screenshot of an example interaction with our game tutor (zoom in to read text more clearly). In this interaction, the player has been presented with a choice of one of three *boons* to augment their build, and asks the game tutor for help in making this choice. The game tutor successfully reads the choices (“Lightning Strike", “Electric Shot", and “Thunder Dash") from the screen and chooses “Lightning Strike". In the following conversation turn, when the game tutor is asked to explain its choice, it shows that the choice was grounded in knowledge of the current build and strategy from the knowledge base. It references the weapon aspect (“Aspect of Eris") and weapon enhancement (“Ricochet Fire") that the player chose previously during the run in its reasoning, and states that it chose “Lightning Strike" because “it's a core boon for the Aspect of Eris." This is in fact a “correct" choice and reasoning, since the widely agreed-upon most powerful build for the Aspect of Eris indeed revolves around Lightning Strike as a core boon.

[1]Has cleared the game on 32 heat with multiple weapons.


B O O N S O F P O S E I D O N
THE SEAS AND ALL THE SURFACE OF THE EARTH BEND TO HIS EVERY WHIM.
CHOOSE ONE:
TIDAL DASH
Your Dash damages foes in an area and knocks them away.
Dash Damage: 35
HYDRAULIC MIGHT
EPIC
Your Attack and Special are stronger the first 10 Sec. in Encounter(s).
Initial Damage Bonus: +75%
OCEAN'S BOUNTY
EPIC
Any , and chamber rewards are worth more.
Reward Value: +60%
Chat Interface
Send

You: Which should I choose?
System: Tidal Dash
You: why?
System: You asked why I chose Tidal Dash over Hydraulic Might and Ocean's Bounty. I chose Tidal Dash because it provides a spammable, high flat damage source that aligns with the build's focus on high damage output. Additionally, it opens up great secondary boons such as Breaking Wave, which can be useful in Styx. Hydraulic Might provides a damage boost to the player's Attack and Special, but it does not provide the same level of spammable damage as Tidal Dash. Ocean's Bounty is not a valid boon in the context of the current build.
You: thanks!
System: You're welcome! It was my pleasure to help you with your Hades build. If you have any more questions or need further assistance, feel free to ask!

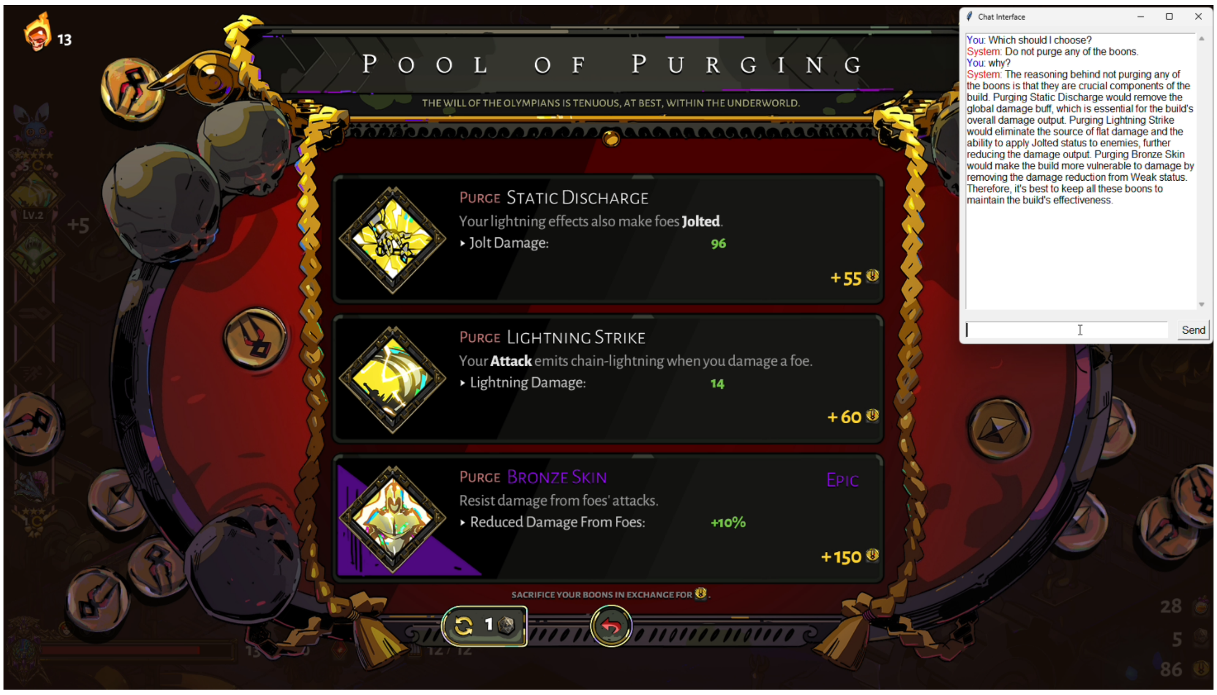
Figure 4: An example interaction with our game tutor. Our system recommends the “correct" boon and grounds its explanation in advanced game knowledge arising from the guides in our knowledge base.

For another example, later on in the run, the player encounters another choice of boons, which we show a screenshot of in Figure 4. The game tutor again successfully reads the choices (“Tidal Dash", “Hydraulic Might", and “Ocean’s Bounty") and recommends “Tidal Dash." When asked to explain its choice, the game tutor again shows appropriate reasoning from knowlege that it has drawn from guides written by advanced players. It says, “I chose Tidal Dash because it provides a spammable, high flat damage source that aligns with the build’s focus on high damage output. Additionally, it opens up great secondary boons such as Breaking Wave, which can be useful in Styx." All of the facts in this explanation are correct; moreover, Tidal Dash is indeed the “correct" choice, as it is in fact considered another core boon for the previously-mentioned powerful Aspect of Eris Build.

In another example interaction, which we show in Figure 5, the player is asked to choose a boon not to add but to remove from their build. Crucially, unlike the choice of adding a boon after an encounter, which is always mandatory, this choice to purge a boon is optional: the player may walk away from the so-called “Pool of Purging" without selecting any of the three options. The game tutor is able to understand the mechanics of the “Pool of Purging" and capable of recommending that the player opt out. Choosing to opt out here is a completely reasonable decision, but there are some weaknesses in this interaction. Namely, there is some hallucination in the reasoning given by the game tutor—several of the facts given about the properties of the boons listed are incorrect. We hypothesize that this may be due to the breadth of knowledge of our knowledge base. While many of the community-written guides discuss which boons to take and for what reason, they generally don’t discuss what boons to purge, since the option to do so is much rarer and it is extremely often correct to simply opt out. Hence, the LLM may lack the ability to justify its choice due to lack of knowledge. In the future, this may be able to be improved with expanding the knowledge base or with more complex chain-of-thought reasoning, for example trying to guide the LLM to weigh the relative value of each boon with the gold sell value.

5 Conclusions & Future Work

Through our work, we have demonstrated the usefulness and feasibility of creating a multi-modal LLM driven game tutor for Hades. Our approach integrates VLMs and RAG to provide context aware and personalized gameplay strategy suggestions. Through understanding the game state and integrating it with expert knowledge, this system is able to deliver meaningful and actionable real-time rec-


POOL OF PURGING
THE WILL OF THE OLYMPIANS IS TENUOUS, AT BEST, WITHIN THE UNDERWORLD.
PURGE STATIC DISCHARGE
Your lightning effects also make foes Jolted.
Jolt Damage: 96
+55
PURGE LIGHTNING STRIKE
Your Attack emits chain-lightning when you damage a foe.
Lightning Damage: 14
+60
PURGE BRONZE SKIN
EPIC
Resist damage from foes' attacks.
Reduced Damage From Foes: +10%
+150
SACRIFICE YOUR BOONS IN EXCHANGE FOR
Chat Interface
Send

You: Which should I choose?
System: Do not purge any of the boons.
You: why?
System: The reasoning behind not purging any of the boons is that they are crucial components of the build. Purging Static Discharge would remove the global damage buff, which is essential for the build's overall damage output. Purging Lightning Strike would eliminate the source of flat damage and the ability to apply Jolted status to enemies, further reducing the damage output. Purging Bronze Skin would make the build more vulnerable to damage by removing the damage reduction from Weak status. Therefore, it's best to keep all these boons to maintain the build's effectiveness.

Figure 5: An example interaction with our game tutor.

ommendations, and shows promise for enhancing player experience and reducing barriers for new players. It bridges the gap between static game guides and interactive assistance, creating an intuitive and accessible support system for gamers of varying skill levels.

Future applications of this work could extend to various gaming genres such as first-person shoots (FPS), massively multiplayer online games (MMOs), and strategy games. These genres pose unique challenges, such as handling dynamic multiplayer environments and coordinating complex team strategies. For instance, in FPS games, the tutor could analyze player positioning and weapon choices to suggest tactical improvements in real-time. In MMOs, it could provide guidance on class-specific skill rotations, resource management, or raid strategies.

Additionally, this work would benefit from integrating more information into the understanding of the game state. For example, future work could involve developing a video-based understanding of the game state for better understanding of fast-paced action in games, as opposed to relying on single frames, and possibly leveraging video guides (e.g. found on YouTube). An additional possible direction is integrating knowledge of player key and mouse inputs throughout gameplay to provide more fine-grained advice on player mechanics.

The intersection of multimodal AI and gaming offers significant potential for innovation, and we envision this work as a step toward creating more intelligent and adaptable virtual game tutors.

Acknowledgments

We express gratitude to Pauline Arnoud, our project mentor, and Jin Zhang and Michael Stopa from Sony for their invaluable guidance and feedback.

References

Adrià Puigdomènech Badia et al. 2020. Agent57: Outperforming the atari human benchmark. *arXiv preprint arXiv:2003.13350*.

Daniele Rege Cambrin, Gabriele Scaffidi Militone, Luca Colomba, Giovanni Malnati, Daniele Apiletti, and Paolo Garza. 2024. Level up your tutorials: Vlms for game tutorials quality assessment. *Preprint*, arXiv:2408.08396.

Victor do Nascimento Silva and Luiz Chaimowicz. 2017. A tutor agent for moba games. *Preprint*, arXiv:1706.02832.

Roberto Gallotta, Graham Todd, Marvin Zammit, Sam Earle, Antonios Liapis, Julian Togelius, and Georgios N. Yannakakis. 2024. Large language models and games: A survey and roadmap. *IEEE Transactions on Games*, page 1–18.

Akshit Ireddy. 2024. Ai plays god of war. `https://github.com/AkshitIreddy/AI-Plays-God-of-War`. GitHub repository.

Omar Khattab, Keshav Santhanam, Xiang Lisa Li, David Hall, Percy Liang, Christopher Potts, and Matei Zaharia. 2022. Demonstrate-search-predict: Composing retrieval and language models for knowledge-intensive NLP. *arXiv preprint arXiv:2212.14024*.

Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan, Saiful Haq, Ashutosh Sharma, Thomas T. Joshi, Hanna Moazam, Heather Miller, Matei Zaharia, and Christopher Potts. 2024. Dspy: Compiling declarative language model calls into self-improving pipelines.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2021. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Preprint*, arXiv:2005.11401.

David Silver et al. 2016. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489.