11-766: Large Language Model Applications Due Thursday, April 30 at 11:59 PM

Project Final Report: SpielBot

Siddhant Joshi *sjoshi2*

1 Abstract

Board game rules disputes are a universal pain point: players frequently encounter edge cases and ambiguous interactions that rulebooks do not clearly address, and general-purpose LLMs hallucinate rules or conflate mechanics across games when prompted without grounding. SpielBot is a domain-specific retrieval-augmented generation (RAG) system that addresses this by grounding its responses in both official rulebooks and community forum threads from BoardGameGeek (BGG)—the internet's largest board game knowledge base—with explicit source citations distinguishing official rules from community guidance. Additionally, SpielBot supports image-based queries through a vision-language model (VLM) pipeline, allowing players to photograph their board mid-game and ask questions with added visual context. The system was built and evaluated over three games chosen for their diversity: Catan (popular, broad coverage), Splendor (small, straightforward rulebook), and Root (niche, complex, asymmetric factions). Evaluation on a custom 30-question benchmark scored by an LLM-as-judge across correctness, completeness, and conciseness shows that SpielBot outperforms off-the-shelf chatbot UIs and approaches dedicated commercial tools, while offering capabilities that no existing solution provides.

2 Introduction

Anyone who plays board games regularly knows the frustration: someone thinks they remember how a rule works, someone else disagrees, and suddenly half the table is passing a rulebook around trying to find the one paragraph that settles it. Even with more experienced groups, many of the questions that actually come up mid-game are edge cases that the rulebook does not clearly address. Nowadays, interactions between multiple rules, ambiguous timing, or player- and scenario-specific exceptions are left up to Google or asking an AI chatbot, but neither are reliable. BoardGameGeek forum threads oftentimes have the answers, but finding the right thread mid-game is tedious. And general-purpose LLMs like ChatGPT or Gemini have no grounding in any specific rulebook, frequently conflating mechanics across games, inventing plausible-sounding rules, or giving confidently wrong answers. In my own testing, Gemini got basic Catan trade rules wrong, and ChatGPT invented card interactions for Root that have no basis in the actual game.

Two community-made products have entered this space. RulesBot.ai [2024] offers a chat interface for roughly 30 curated games, but its catalog is small, it is text-only, and in my testing responses were largely vague or incorrect—most likely because each game bot is trained on a (potentially outdated) static copy of a the official rulebook with no community data. Board Game Wizard [2024] takes a more consumer-facing approach with voice-activated rules lookup and a seemingly broader game library, but it similarly relies only on official rulebooks and offers no image-based query

support. Neither tool incorporates the community-vetted edge case rulings that players actually need, and neither can interpret a photo of the current board state to provide scenario-specific feedback.

SpielBot is built to fill these gaps. It is a domain-specific RAG system that grounds its responses in two complementary data sources: official rulebooks and community forum threads scraped from BoardGameGeek, the de facto hub for board game information with forums spanning over 100,000 games. Every response explicitly cites whether it draws from official rules or community guidance, so the player can judge the authority of the answer. Beyond text, SpielBot supports image-based queries through a vision-language model (VLM) pipeline: a player can photograph their board mid-game, ask a question, and the system will interpret the visual context alongside the text query before retrieving and generating an answer. This combination of dual-source retrieval with citations and multimodal input support is something no existing tool provides.

The system was built and evaluated over three games chosen to stress-test different aspects of the pipeline. **Catan** is one of the most popular modern board games, with extensive internet coverage and a well-structured rulebook. **Splendor** has a small, straightforward rulebook with few edge cases, testing how the system handles simple look-ups without over-complicating them. **Root** is a niche, asymmetric game where each of twelve factions plays by fundamentally different rules and 139 unique cards across multiple decks, making it the hardest test of chunking, retrieval, and generation.

This project engages directly with several topics from the course: end-to-end RAG pipeline construction (data ingestion, chunking, embedding, hybrid retrieval, generation), prompt engineering for grounded generation and chain-of-thought reasoning, LLM-as-judge evaluation at scale, and exploration of VLM capabilities for multimodal input. The evaluation framework uses a custom 30-question benchmark with retrieval annotations, scored across nine experiment settings including chatbot UIs, API baselines, and commercial competitors. It was designed to quantitatively demonstrate whether a purpose-built RAG system can outperform the tools an everyday board gamer would actually reach for mid-game.

3 Literature Review

The technical challenges underlying SpielBot span several areas: evaluating how well LLMs follow complex rule systems, building effective RAG pipelines over PDF documents, improving retrieval coverage through multi-query strategies, and leveraging vision-language models for multimodal input. I will present a review of the literature in these spaces, organized by topic.

3.1 Rule-Following and Board Game Comprehension

Zhou et al. [2024] introduce RuleArena, a benchmark designed to evaluate LLMs' ability to follow complex, real-world rules expressed in natural language. They test on airline baggage fees, NBA transaction regulations, and tax policies, finding that LLMs frequently struggle to identify the correct rule when similar but distinct regulations are present and that rules requiring the aggregation of multiple intermediate results are particularly challenging. The authors measure rule-wise recall (whether the model identifies the correct rule(s)), application correctness (whether

it applies the rule correctly), and problem accuracy (overall correctness). Board games are full of exactly this kind of interacting-rule complexity. For instance, a single turn in Root can involve movement restrictions, battle rules, item exhaustion, and faction-specific abilities all interacting at once. RuleArena's evaluation framework, which measures rule-wise recall, application correctness, and overall problem accuracy, also provided a useful template for thinking about SpielBot's own evaluation metrics.

Becker et al. [2025] present Boardwalk, a framework for evaluating whether LLMs can implement working digital versions of board games from natural language rules. To isolate genuine comprehension from memorization, the authors anonymize all 12 test games, giving them fake names and renaming pieces and concepts. The best-performing model, Claude 3.7 Sonnet, produced error-free implementations only 55.6% of the time. An interesting finding was that the most commonly failed games were those using graph-based boards with rules requiring reasoning about adjacency and spatial positioning. Considering LLMs struggle to reason about board geometry from clean textual descriptions, interpreting spatial relationships from user-uploaded photos for SpielBot's VLM component demanded careful architectural consideration during development.

3.2 RAG over PDF Documents

Khan et al. [2024] provide a practitioner-oriented study of constructing RAG pipelines that ingest PDF documents. They explore chunking strategies, embedding model selection, and retrieval methods, concluding that semantic chunking outperforms naive page-boundary splits and that a hybrid combination of dense vector search with sparse BM25 retrieval outperforms either method alone. They also observe that rulebook-style PDFs (with diagrams, cross-references, and multi-column layouts) present unique parsing challenges that require specialized pre-processing. Both of these findings directly informed SpielBot's design: I adopted semantic chunking based on section markers and sub-topic boundaries rather than fixed token counts, and implemented hybrid dense + BM25 retrieval with Reciprocal Rank Fusion (RRF). Their evaluation framework, which measures retrieval precision alongside end-to-end answer quality using LLM judges, also served as a template for SpielBot's evaluation pipeline.

3.3 Multi-Query Retrieval

Li et al. [2024] introduce DMQR-RAG, a Diverse Multi-Query Rewriting framework that addresses the narrow retrieval scope of single queries. They observe that standard single-query rewriting often fails to retrieve all relevant documents, and that even naive multi-query approaches like RAG-Fusion tend to produce rewrites that are too similar to one another, yielding redundant retrieval results with low recall of genuinely relevant documents. DMQR-RAG proposes four rewriting strategies that operate at different levels of information to maximize the diversity of retrieved documents. They also introduce an adaptive strategy selection method that identifies the most suitable rewriting approach for each query, achieving better retrieval and response quality with fewer total rewrites. Specifically, DMQR-RAG's "least-to-most" decomposition strategy largely informs the reasoning-driven pipeline in SpielBot, resulting in some great performance gains on later iterations. While it does not exactly mirror it, the core insight of diverse queries improving chunk capture remains.

3.4 Vision-Language Models

Bordes et al. [2024] survey the landscape of vision-language model architectures, covering contrastive models like CLIP, generative models like LLaVA and GPT-4V, and hybrid approaches. The evaluation benchmarks they discuss include VQA, GQA, and MMBench, which test spatial reasoning and visual question answering skills (something necessary for interpreting board game photos). They also discuss known VLM weaknesses, most notably that current models lag behind on reasoning about spatial relationships between objects in a scene. This implied that SpielBot's image pipeline could not simply pass a photo to a VLM and expect an accurate game-state interpretation; instead, the system uses a two-stage decoupled architecture where the VLM produces a structured textual scene description that is then fed into the standard RAG pipeline, keeping the visual understanding and rule retrieval concerns cleanly separated.

3.5 Industry Landscape

The non-academic landscape for this problem remains largely untapped, with multiple online posts in board game communities and a few prototypes made by hobbyists. RulesBot.ai [2024] offers a chat interface for approximately 30 curated games, but its small catalog, text-only interface, and lack of community data result in responses that are often vague or unhelpful. Board Game Wizard [2024] takes a more consumer-facing approach with voice-activated lookup and a broader game library, but similarly relies only on official rulebooks. Neither tool incorporates community forum data or supports image-based queries. BoardGameGeek [2025] itself, while not a question-answering system, represents the richest source of board game community knowledge on the internet, with dedicated rules forums for virtually every published game. The gap between the community knowledge sitting in BGG forums and the tools available to access it during a game is precisely what SpielBot targets.

4 Method

4.1 Data Collection and Ingestion

SpielBot's data comes from three sources: official rulebook PDFs, community forum threads from BoardGameGeek, and structured card data. Each required a different ingestion approach.

Rulebook PDFs. Rulebook text was extracted using PDFPlumber via a custom `pdf_extractor.py` script. Board game rulebooks turned out to be surprisingly difficult to reliably parse dur to many factors. Primarily, their multi-column layouts caused interleaved sentence fragments, page illustrations bled map coordinates and dice-pip symbols into the text stream, and PDF watermark strings appeared as literal text. I handled some of the systematic garbling with manual patch files, but ended up largely processing the data manually as it was just three games to prepare. Rulebook chunks are tagged with `source_type:"rulebook"` to distinguish them from other data in retrieval.

BGG Forum Threads. Forum data was scraped from the rules sub-forums for each game using the BGG XMLAPI2. Each thread was processed through an LLM to extract structured {`question, answer, confidence`} tuples. Each forum chunk also carries a `confidence` field (`high`, `medium`, or `low`) derived from whether a publisher representative responded, whether the thread reached

consensus, and the level of disagreement among respondents. This pipeline uses the following asymmetric embedding design: the `embed_text` field stores the original question from the thread, while the `content` field stores the LLM-extracted summary and resolution. The generator system prompt uses this field to calibrate how confidently it presents community-sourced information. This is different from rulebook retrieval, because the matching criteria and retrieved chunk are different. With rulebooks, the matching is done over the same information that is to be retrieved whereas with forum retrieval, the matching is done over the question topics and retrieved information is the discussion that took place in response. Forum chunks are tagged with `source_type:"forum"` to distinguish them from other data in retrieval.

Root Card Data. Root has 139 unique cards across multiple decks, each with a Suit, Cost, and Effect. I compiled these into a structured CSV and processed them into retrieval-ready chunks, grouped by card name with deck, effect, and cost metadata. Card chunks are tagged with `source_type:"card"` to distinguish them from other data in retrieval.

4.2 Chunking Strategy

Chunking quality turned out to be one of the more impactful design decisions in the system. The SpielBot chunking strategy uses rulebook structure-aware chunking. Sections with explicit title markers are never merged with adjacent sections, even if short, in order to represent distinct rulebook topics that should remain separate retrieval units. However, when sections exceed 500 tokens, they are split and continued in a new chunk. Footers and callout phrases are included in their respective sections rather than floating as standalone chunks to ensure a retrieval hit on a rule also surfaces its exceptions and conditions.

To further improve retrieval quality, section titles are prepended to the embedded text, which proved critical for closing vocabulary gap failures. Second, overlap was eliminated entirely as each chunk already started at a natural topic boundary, so the retriever handles cross-chunk answers by returning multiple relevant chunks rather than relying on duplicated tail text.

For forum chunks, the "unit" is a complete QA pair, preserving the question-answer semantics that arbitrary mid-conversation splitting would destroy. Chunk size constants were set to be between 50 and 400 tokens to keep generator context focused when 6-8 chunks are retrieved simultaneously.

4.3 Indexing and Retrieval

The retrieval pipeline uses a hybrid approach combining dense and sparse search, following the finding from Khan et al. [2024] that the combination outperforms either method alone. Dense retrieval uses `BAAI/bge-base-en-v1.5` embeddings stored in ChromaDB, with game-based tagging ensuring a Catan question never retrieves Root chunks. Sparse retrieval uses BM25 via the `rank-bm25` Python library, with per-game in-memory indexes built from the same chunk corpus. The stopword list is kept relatively conservative, preserving words like "may," "must," and "not" that carry critical semantic weight in rules text. Hybrid fusion combines dense and sparse candidates via Reciprocal Rank Fusion (RRF) with $k = 60$, where each chunk's fused score is computed as:

$$\sum_{s \in \{\text{dense,sparse}\}} \frac{1}{k + \text{rank}_s(c)}$$

A critical architectural decision was splitting the retrieval vectorstores into two independent pools. An early version used a single shared ChromaDB collection for all chunk types, but during testing I observed that rulebook chunks consistently outranked forum chunks in the merged retrieval. Rulebook text is more keyword-rich than conversational forum posts, giving it a systematic advantage in both dense and sparse scoring. The solution was to separate the data into two ChromaDB collections, one for rulebook/card chunks and one for forum chunks, each with its own BM25 index. Each collection has separate retrieval budgets of 5 chunks from the rulebook/card pool and 3 from the forum pool to ensure retrieval diversity.

4.4 Reasoning Pipeline

A reasoning-driven pipeline was developed to enhance retrieval quality and breadth.. Inspired by the multi-query rewriting approach of DMQR-RAG (Li et al. [2024]), the pipeline uses a stronger reasoning model (Gemini 2.5 Pro) to decompose the user's query into sub-queries, each of which is used as a separate retrieval query. The results are deduplicated via a union of the chunks to produce an expanded context set that covers more of the relevant rule space than any single query could.

4.5 Vision Pipeline

SpielBot supports image-based queries through a two-stage decoupled architecture. In the first stage, a VLM (which is the same as the reasoning model) analyzes an uploaded game-state photo and produces a rich textual description of the board—piece positions, card layouts, resource counts, and visible tokens. Game-specific scene analysis prompts guide the VLM to focus on the components relevant to each game, for example, hex terrain types and settlement positions for Catan. In the second stage, this scene description is appended to the user's text query and fed into the standard RAG pipeline. A conditional instruction in the generator's base system prompt activates only when a scene description is present, instructing the generator to reference the board state naturally in its answer. This also help steer generic answers to be formulated with the user's unique game-state in mind. This decoupled design keeps the vision component cleanly separated from retrieval, making both independently testable, and avoids the complexity of end-to-end visual question answering, This is exactly what Becker et al. [2025] and Bordes et al. [2024] note current models struggle with for the spatial reasoning board games require.

4.6 Generation

The generator is accessed via a LiteLLM proxy, with Gemini 2.5 Flash as the primary generation model. Per-game system prompts are maintained, with each one emphasizing common confusions and terms of a game. The base system prompt instructs the model to prioritize official rulebook citations over community posts, use chain-of-thought reasoning for multi-rule interactions, cite sources explicitly, and keep responses concise. The user message also includes the number of retrieved chunks and a note that sources are excerpts rather than the complete rulebook, giving the model explicit awareness of its limited context. As mentioned earlier, in the presence of a scene description, a conditional instruction in the generator's base system prompt activates to steer answers to be generated with respect to the user's game-state.

4.7 Evaluation Scheme

Evaluation was a substantial engineering effort in its own right. No standard benchmark exists for board game rules Q&A, so I built a custom evaluation pipeline from scratch, covering both retrieval quality and end-to-end answer quality.

Evaluation Dataset. I created a dataset of 30 question-answer pairs: 10 per game, split evenly between comprehension questions (single-lookup rule clarifications, e.g., "How do you win a noble in Splendor?") and reasoning questions (multi-rule interactions and edge cases, e.g., "As the Scoundrel vagabond, can I Scorched Earth the clearing with the Marquise's Keep?"). Each question was written by hand based on real rules disputes I have encountered, then annotated with a gold-standard reference answer. The distinction between comprehension and reasoning questions tests two fundamentally different retrieval challenges. Comprehension questions typically require one chunk look-ups, while reasoning questions may require synthesizing information in 2-4 chunks from different rulebook sections. Each question was also annotated with the specific chunk IDs that contain the answer so I could conduct retrieval-only evaluation independent of generation quality.

Retrieval Metrics. The retriever always fills its per-query budget of 8 chunks, but questions have varying numbers of relevant chunks, so I used adaptive metrics that do not penalize fixed retrieval budgets. The primary metrics are:

- Mean Average Precision (MAP): Adapts to the number of relevant chunks per question
- Normalized Discounted Cumulative Gain (NDCG@8): Applied logarithmic discounting so a relevant chunk at position 1 contributes far more than one at position 7
- Mean Reciprocal Rank (MRR):
- R-Precision: Precision@R where $R = |\text{relevant}|$; top-R slice

Recall@8 and Hit Rate@k are reported as secondary metrics.

Answer Quality. Overall answer quality was evaluated using an LLM-as-judge framework with Llama 3.3 70B (via Groq) as the judge model. Each answer is scored on a 1-5 scale across three dimensions: *correctness* (does the answer state the right rule?), *completeness* (does the answer cover all relevant rules and edge cases?), and *conciseness* (is the answer free of hallucinated or extraneous information?). The composite score is the mean of all three. The judge receives the question, the gold reference answer, and the model's response without any indication of which setting produced it. I manually spot-checked 3-5 scored answers per setting to validate judge calibration and verify that the rubric was not systematically favoring irrelevant answer features, like verbosity.

Comparison Settings. Answers were collected across nine evaluation settings designed to span the range of tools a board gamer might actually reach for mid-game:

- Chatbot UI baselines collected by manually entering each question into ChatGPT, Claude, Gemini, and Perplexity
- GPT-5 API with a rules-expert system prompt but no game data
- GPT-5 API with the same system prompt plus the full rulebook PDF attached via OpenAI's file input API

- Board Game Wizard and Rulesbot.ai
- SpielBot

The chatbot UI responses were collected manually to reflect what a real user would experience.

5 Experimental Results

This section describes the key experiments and iterative design decisions that shaped the final version of SpielBot, organized around the three evaluation axes: retrieval quality, overall answer quality, and failure analysis.

5.1 Retrieval Evaluation

Baseline Results

The first retrieval evaluation was run on the initial pipeline of a single ChromaDB collection, `bge-small-en-v1.5` embeddings, no re-ranker, and no source-type split. Table 1 shows the results.

Metric	Overall	Catan	Root	Splendor
MRR	0.541	0.575	0.573	0.475
MAP	0.271	0.248	0.281	0.283
R-Precision	0.269	0.233	0.308	0.267
NDCG@8	0.378	0.346	0.396	0.392
Recall@8	0.417	0.325	0.442	0.483
Hit Rate@8	0.833	0.700	0.900	0.900

Table 1: Baseline retrieval metrics, overall and by game.

Right away, I noticed several concerns. An MRR of 0.541 meant the first relevant chunk appeared at roughly position 2 on average. More worrying was MAP at 0.271 and Recall@8 at 0.417, meaning the system was retrieving fewer than half of the relevant chunks and ranking them poorly when it did find them. Hit Rate@8 of 0.833 meant 5 of the 30 questions returned *zero* relevant chunks in the top-8, which is a complete retrieval failure! Breaking down the metrics based on question type, (Table 2) I noticed some expected failure modes. Reasoning questions had higher MRR (0.627 vs. 0.456) but lower Recall@8 (0.372 vs. 0.461). This meant for complex multi-rule questions, the system was good at surfacing *one* relevant chunk but struggled to find *all* the relevant chunks. Reasoning questions did achieve perfect Hit Rate@8 (1.000) while comprehension lagged at 0.667. This suggested the system's retrieval was biasing toward partial coverage of complex topics rather than complete coverage of any one topic.

Metric	Comprehension (n = 15)	Reasoning (n = 15)
MRR	0.456	0.627
MAP	0.298	0.243
Recall@8	0.461	0.372
Hit Rate@3	0.533	0.733
Hit Rate@8	0.667	1.000

Table 2: Baseline retrieval metrics by question type.

Identifying Improvements

With some further analysis of the errors on the baseline results, I identified a few specific problems and their respective fixes for performance enhancement:

1. **Forum chunks drowned by rulebook chunks**: As mentioned earlier, in the single-collection design, rulebook text systematically outranked forum chunks in both dense and sparse scoring. To address this, I split it into two independent ChromaDB collections (rulebook/card and forum), each with its own BM25 index, and enforcing separate retrieval budgets (5 rulebook + 3 forum).
2. **Vocabulary gap failures**: Certain queries exposed a potential needle-in-a-haystack issue, where relevant chunks were missed for not containing keywords in their content. For example, a query about "corvid plots" in Root missed a chunk titled "13.7 Plot Tokens Reference" that listed the plot effects without ever using the word "corvid." The fix was prepending section titles to the embedded text, bringing the title vocabulary into the embedding space.
3. **Embedding model capacity**: The lightweight `bge-small-en-v1.5` (33M parameters) was upgraded to `bge-base-en-v1.5` (110M parameters) for richer semantic representations.
4. **No re-ranking**: A cross-encoder re-ranker (`cross-encoder/ms-marco-MiniLM-L-6-v2`) was added to refine ranking quality within each retrieval pool after the initial hybrid fusion.
5. **Multi-query retrieval**: Perhaps the most important retrieval improvement was having a reasoning model decompose queries into finer sub-queries that casted a wider net over the chunks and would end up pulling more relevant information than a single lookup.

Improved Results

After applying these improvements, retrieval quality improved substantially across every metric. Table 3 and Figure 1 show the final results in different formats.

MRR rose from 0.541 to 0.671, meaning the first relevant chunk moved from approximately position 2 to position 1.5 on average (a positive change!). MAP nearly doubled from 0.271 to 0.422, reflecting much better ranking of all relevant chunks. Recall@8 improved from 0.417 to 0.663, so the system now retrieves roughly two-thirds of relevant chunks, up from fewer than half. What was most exciting, was Hit Rate@8 climbed from 0.833 to 0.933, reducing complete retrieval failures from 5 questions to 2!

Table 3: Final retrieval metrics after all improvements, overall and by game.

Metric	Overall	Catan	Root	Splendor
MRR	**0.671**	0.600	0.678	0.733
MAP	**0.422**	0.342	0.399	0.525
R-Precision	**0.400**	0.425	0.392	0.383
NDCG@8	**0.558**	0.496	0.518	0.660
Recall@8	**0.663**	0.615	0.575	0.800
Hit Rate@8	**0.933**	1.000	0.900	0.900

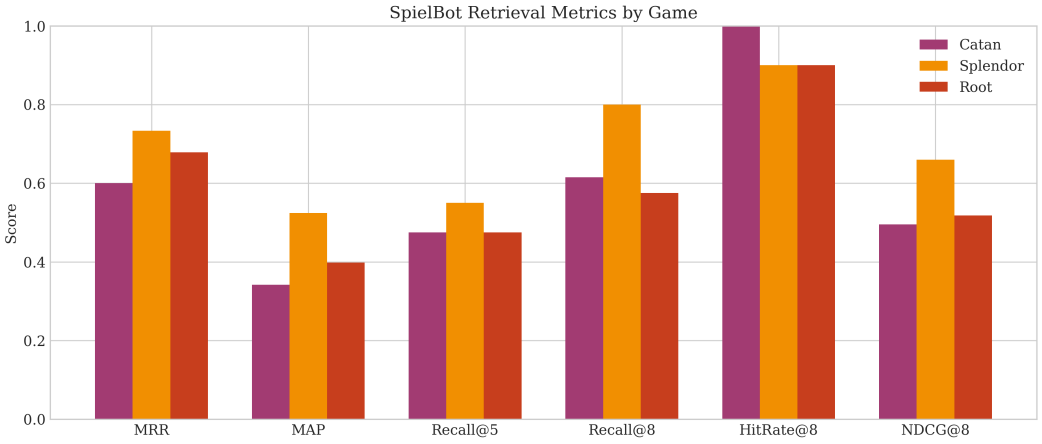

Figure 1: SpielBot retrieval quality across adaptive metrics (final system).

Per-game, Splendor achieved the highest scores across most metrics (MRR 0.733, Recall@8 0.800), consistent with its small focused rulebook. Catan improved the most dramatically, going from the worst Hit Rate@8 (0.700) to perfect (1.000). Unfortunately, Root had the lowest Recall@8 (0.575), which is understandable given the complexity of the game, but managed to maintain a Hit Rate@8 of 0.900.

The comprehension vs. reasoning asymmetry persisted in the improved system (reasoning MRR 0.708 vs. comprehension 0.633; reasoning Recall@8 0.639 vs. comprehension 0.688), confirming that the gap is structural rather than an artifact of the baseline configuration. Reasoning questions continued to achieve perfect Hit Rate@8 (1.000) while comprehension improved from 0.667 to 0.867. This consistent pattern—strong first-hit performance but incomplete multi-chunk coverage on complex queries—is exactly what the multi-query reasoning pipeline targets.

5.2 Answer Quality Evaluation

Table 4 shows the LLM-as-judge results across all nine evaluation settings. Each score is the mean across 30 questions on a 1-5 scale; the composite is the average across correctness, completeness, and conciseness.

Setting	Correct	Complete	Concise	Composite
GPT-5 + PDF	4.67	4.57	3.73	**4.32**
Board Game Wizard	3.90	3.80	4.13	3.94
GPT-5 + Prompt	4.07	3.97	3.60	3.88
Perplexity (UI)	4.33	4.43	2.20	3.66
Claude (UI)	3.80	3.97	3.10	3.62
SpielBot	**3.80**	**3.47**	**3.50**	**3.59**
Gemini (UI)	3.93	3.97	2.60	3.50
ChatGPT (UI)	3.40	3.53	3.00	3.31
RulesBot.ai	2.20	2.23	1.97	2.13

Table 4: Answer quality by evaluation setting (1-5 scale, higher is better).

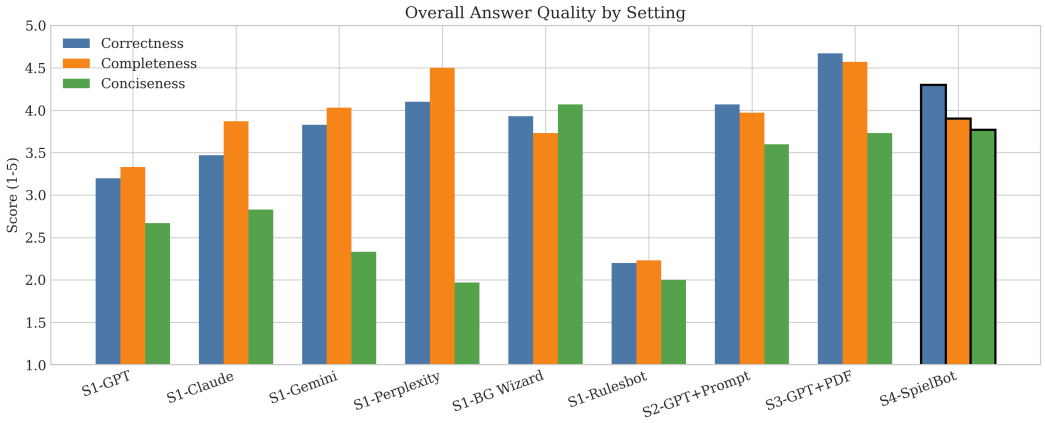

Figure 2: Answer quality across evaluation settings, broken down by correctness, completeness, and conciseness.

GPT-5 API with the full rulebook PDF was the strongest setting, which is expected given it has the complete rulebook in context. Among the chatbot UI baselines, Perplexity scored highest on correctness and completeness, but was heavily penalized on conciseness due to verbose, often padded responses. Gemini produced surprisingly poor answers on some basic questions, getting basic Catan trade rules wrong despite being a strong general-purpose model. In terms of existing board game assistants, RulesBot.ai performed worst across all dimensions, primarily driven by the fact that Splendor was not covered in its game selection and that most responses were vague or incorrect.

SpielBot outperformed all the general chatbot UIs and RulesBot.ai, while barely trailing Board Game Wizard and GPT-5 + PDF. Notably, SpielBot's conciseness score was the second-highest among all settings after Board Game Wizard, reflecting the system prompt's emphasis on focused, source-grounded responses.

5.3 Comprehension vs. Reasoning Breakdown

Figure 3: Comprehension vs. reasoning correctness across evaluation settings.

Reasoning questions received systematically lower scores across all settings, which aligns with the expectation that synthesizing information over multiple rules is a harder task than comprehension. SpielBot's relative performance on reasoning questions was closer to the strongest baselines than on comprehension, suggesting that forum data was providing genuine value that rulebook-only systems miss.

6 End-of-Semester Goals and Next Steps

I met all my end-of-semester goals outlined in my proposal! I think there are many additional avenues I would like to take the project down and will definitely be exploring them in the summer. To begin, I want to put iron out a scalable rulebook extraction-chunking pipeline to be able to serve more games, faster. I would also like to experiment with smarter chunking strategies that would make chunks slightly more focused so that larger chunks don't run into the needle-in-a-haystack problem during retrieval time. There is some fascinating potential in the dataset for this problem, where I can expand the reasoning and comprehension questions to operate in general board game rule spaces and also explore scenario-based evaluation that can experiment with providing unique play suggestions.

7 Use of AI

I used AI to format graphics and tables in LaTeX and to help with some analysis of my experimental metrics.

References

Adam G. Becker, Rachel Beddor, Avery Lamp, Julian Matus, Natalie Maus, and Finbarr Timbers. Boardwalk: Towards a framework for creating board games with LLMs. *arXiv preprint arXiv:2508.16447*, 2025. URL https://arxiv.org/abs/2508.16447.

Board Game Wizard. Board Game Wizard — voice-activated rules lookup, 2024. URL https://www.board-game-wizard.com. Accessed: 2026-04-28.

BoardGameGeek. BoardGameGeek — the board game community, 2025. URL https://boardgamegeek.com/. Accessed: 2026-04-28.

Florian Bordes, Richard Yuanzhe Pang, Anurag Ajay, Alexander C. Li, Adrien Bardes, Suzanne Petryk, Oscar Mañas, Zhiqiu Lin, Anas Mahmoud, Bargav Jayaraman, Mark Ibrahim, Diana Hall, Yunyang Xiong, Jonathan Lebensold, Candace Ross, Srihari Jayakumar, Chuan Guo, and Diane Bouchacourt. An introduction to vision-language models. *arXiv preprint arXiv:2405.17247*, 2024. URL https://arxiv.org/abs/2405.17247.

Adnan Ali Khan, Usman Shahzad, Sadaf Ilyas, Seemab Latif, and Rabia Latif. Developing retrieval augmented generation (RAG) based LLM systems from PDFs: An experience report. *arXiv preprint arXiv:2410.15944*, 2024. URL https://arxiv.org/abs/2410.15944.

Zhicong Li, Jiahao Wang, Zhishu Jiang, Hangyu Mao, Zhongxia Chen, Jiazhen Du, Yuanxing Zhang, Fuzheng Zhang, Di Zhang, and Yong Liu. DMQR-RAG: Diverse multi-query rewriting for retrieval-augmented generation. *arXiv preprint arXiv:2411.13154*, 2024. URL https://arxiv.org/abs/2411.13154.

RulesBot.ai. RulesBot.ai — board game rules chat assistant, 2024. URL https://www.rulesbot.ai/chat/. Accessed: 2026-04-28.

Ruiwen Zhou, Wenyue Hua, Liangming Pan, Sitao Cheng, Xiaobao Wu, En Yu, and William Yang Wang. RuleArena: A benchmark for rule-guided reasoning with LLMs in real-world scenarios. *arXiv preprint arXiv:2412.08972*, 2024. URL https://arxiv.org/abs/2412.08972.