LLM-powered Fantasy Football Drafter

Dhruv Chauhan and Tanay Kommareddi

Introduction

Every year, tens of millions of people across the country play fantasy football. The goal is simple - draft a team that scores the most points over the course of a season in a league with your friends, where everyone is picking from the same pool of players. While it might sound straightforward, there's a lot of strategy involved. You have to decide what positions to prioritize, whether a player has a good supporting cast, if they've recently switched teams, what their schedule looks like, and more.

Formally, the standard fantasy roster includes a quarterback (QB), two wide receivers (WR), two running backs (RB), a tight end (TE), a FLEX (which can be a WR, RB, or TE), a kicker, a defense, and a bench (which consists of whichever kinds of players you want). For our project, we focused on the core of a team and simplified the format to: one quarterback, three wide receivers (because 1 receiver can account for the FLEX position), two running backs, and one tight end.

To tackle this, we built a hierarchical multi-agent LLM-based drafter using AutoGen. Our system includes a head agent that coordinates with specialized position agents, where each is responsible for extracting and analyzing data about players at their position, before making a final draft decision. During a simulated draft, the agent is given a list of available players and builds a complete team that fits the roster format, all while reasoning over team needs and player performance.

Ultimately, we evaluate the agent's performance by comparing its drafted teams to two benchmarks: a league average based on greedy drafting using ESPN pre-season rankings, and an ideal team composed of the top 7 projected players from ESPN that fit the required slots, without any draft constraints.

The full implementation, including code and tools, can be found *here*.

Methods

As mentioned in the introduction, we use a hierarchical multi-agent architecture to draft a fantasy football team. To implement this, we use Microsoft's AutoGen framework, which enables a structured "group chat" between agents, each powered by GPT-4o-mini.

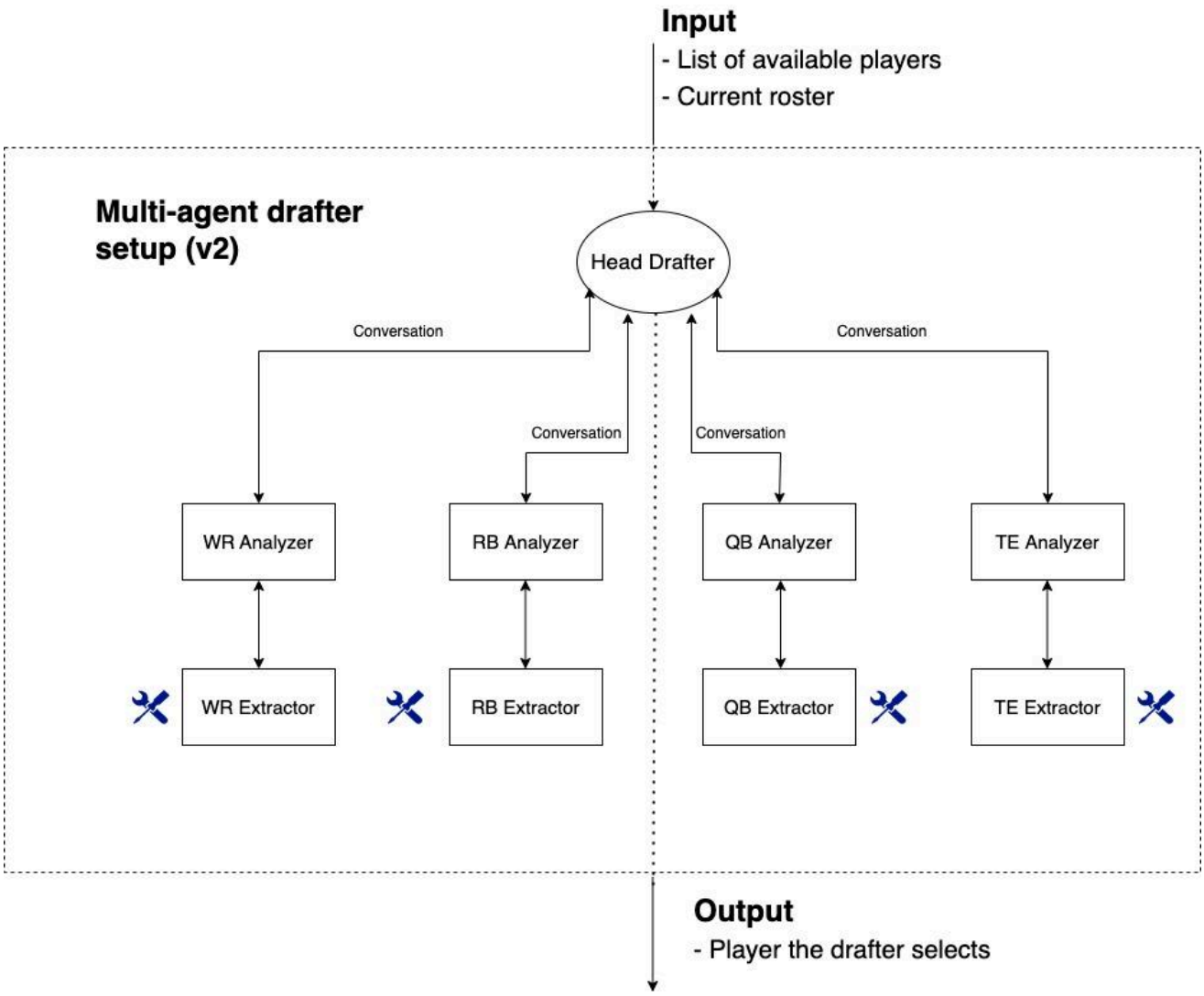
The conversation is coordinated by a Group Chat Manager, which helps orchestrate the flow of the conversation. It ensures that agents speak in the correct order, that messages are routed properly, and that the overall interaction stays coherent.

To support smooth execution of the draft, we also developed a custom drafting software interface. This allowed the LLM agents to interact with the draft environment directly - receiving the list of available players, tracking drafted players across all teams, and making selections automatically. This setup made it easy to run full simulations without manual intervention.

Our multi-agent setup consists of 9 agents:

- One Head Drafter Agent
- Two agents per position - a Position Analyzer and a Position Extractor for QB, RB, WR, and TE positions

An example of a complete conversation flow can be found *here*, based on one of our recorded draft logs.



Group Chat Manager

The Group Chat Manager is a specialized agent provided by the AutoGen framework, designed specifically to manage the flow of a multi-agent conversation. To ensure the conversation followed the structure we intended, we took two key steps:

- We gave the manager a detailed system prompt describing the full turn-by-turn flow of the draft conversation.
- We explicitly defined “allowed communication transitions” between agents to restrict who can talk to whom.

Here is what the system prompt looked like:

```
group_chat_manager_prompt = """
You are a group chat manager for a fantasy football draft conversation. The
head_drafter_agent should ALWAYS start the conversation.
Follow this exact sequence:

1. The head_drafter_agent first analyzes roster needs and identifies positions to fill
2. The head_drafter_agent asks a specific position analyzer for a recommendation
3. The position analyzer then talks next and asks the respective position extractor for data
on the players it gives to the extractor
4. The respective position extractor provides data
5. The position analyzer gives ONE recommendation back to the head_drafter_agent for that
position
6. Steps 2-5 repeat for other NEEDED positions
7. Once the head_drafter_agent has all recommendations from the required position analyzers,
it makes the final selection

The head_drafter_agent should always first analyze what positions to fill and then call on
the appropriate position analyzers for recommendations.

Once you see a **TERMINATE** in ANY message, it is IMPERATIVE that head_drafter_agent should
be the next speaker.

"""
```

To enforce agent communication restrictions, we used the following transition rules:

```
allowed_transitions = {
   head_drafter_agent: [
       qb_analyzer,
       rb_analyzer,
       wr_analyzer,
       te_analyzer,
       head_drafter_agent,
   ],
   qb_analyzer: [qb_extractor, head_drafter_agent],
```

```
   wr_analyzer: [wr_extractor, head_drafter_agent],
   rb_analyzer: [rb_extractor, head_drafter_agent],
   te_analyzer: [te_extractor, head_drafter_agent],
   qb_extractor: [qb_analyzer],
   wr_extractor: [wr_analyzer],
   rb_extractor: [rb_analyzer],
   te_extractor: [te_analyzer],
}
```

To summarize the rules, the head drafter can only speak to analyzer agents (or itself to have multi-step reasoning). The analyzers could only talk to the head drafter and their respective position extractor agents, and the extractors could only talk to their respective analyzer agents.

Head Drafter Agent

The Head Drafter Agent serves as the main point of contact with the drafting software we built. At each turn, it receives a prompt with its current team and a list of players already selected by other teams. Based on this information, it determines who to draft next. The drafting process follows three main steps:

1. Analyze the current roster and identify which positions still need to be filled.
2. Query each relevant Position Analyzer for the best available player at their position.
3. Select a player from the ones recommended based on team needs and overall value.

To ensure the Head Drafter understood its role and behaved predictably, we provided a carefully engineered system prompt. This prompt clearly outlined the agent's responsibilities, encouraged step-by-step reasoning, and enforced a structured output format to support automated draft simulations.

Here is an example of the prompt it received each round:

```
Your current roster:
- Jahmyr Gibbs - RB (DET)


Already drafted players (DO NOT SELECT THESE):
- Christian McCaffrey
- Bijan Robinson
- Breece Hall
- CeeDee Lamb
- Tyreek Hill
- Ja'Marr Chase
```

```
- Amon-Ra St. Brown
- Justin Jefferson
- Jonathan Taylor
- Saquon Barkley
- A.J. Brown
- Jahmyr Gibbs


You are making a selection for the AI_General_Manager. Remember that a complete roster
requires: 1 QB, 2 RB, 3 WR, and 1 TE
First, ask yourself what your current roster is and based on that clearly write out what
positions still have empty slots in.
Based on the empty slots, make a decision on who you want to draft next.
```

Here is the system prompt for the head drafter agent:

```
head_drafter_prompt = """
You are the Head Drafter for a fantasy football team. Your role is to coordinate the draft
process and make the final player selection.
You are in a group chat with position anaylzers, one each for quarterback (QB), wide
receiver (WR), running back (RB), and tight end (TE).
You will ask each position-specific analyzer for their recommendation one at a time.
Make smart decisions based on the recommendations from the position-specific analyzers
answers ONLY and GOOD drafting strategies you already know.
ADHERE to the team roster format which is 1 QUARTERBACK, 2 RUNNING BACKS, 3 WIDE RECEIVERS,
1 TIGHT END
Before asking for recommendations, first list out your current roster. Then based on that,
determine what positions still have empty slots.
Only ask an analyzer if the position is still needed. If a position is already fully filled,
you MUST NOT ask for a recommendation for that position.

DRAFT PROCESS:
1. FIRST: Review the current roster and identify positions that still need to be filled
   - A complete roster needs: 1 QB, 2 RB, 3 WR, 1 TE
   - IMPORTANT: YOUR ROSTER MUST HAVE THIS FORMAT (1 QUARTERBACK, 2 RUNNING BACKS, 3 WIDE
RECEIVERS, 1 TIGHT END
   - IF YOU DON'T HAVE A TIGHT END IN THE LAST ROUND, YOU MUST TAKE A TIGHT END
   - IF YOU DON'T HAVE A QUARTERBACK IN THE LAST ROUND, YOU MUST TAKE A QUARTERBACK
   - You just need recommendations for each position ONCE per round and choose only ONE
player, not the whole roster.

2. SECOND: Request recommendations from position-specific analyzer agents
   - ONLY ask analyzers for positions you still need to fill - do NOT ask extractors for
data
   - Ask one analyzer at a time and wait for their response before asking another
   - NEVER make up or predict what an analyzer might recommend - wait for their actual
response

3. THIRD: After collecting recommendations, evaluate them and select ONE player to draft

   - In early rounds, prioritize RB and WR positions unless an exceptional QB or TE is
```

```
available
   - Never consider players from the "already drafted" list
   - Try not to draft players on the same NFL team

COMMUNICATE CLEARLY:
- When asking an analyzer for a recommendation, direct your message specifically to them
(e.g., "running_back_analyzer, what is your recommendation?")
- Wait for each analyzer to respond before asking the next one
- When making your final decision, clearly state the selected player's full name

FINAL SELECTION FORMAT:
- IMPORTANT: YOUR ROSTER MUST HAVE THIS FORMAT (1 QUARTERBACK, 2 RUNNING BACKS, 3 WIDE
RECEIVERS, 1 TIGHT END
- IF YOU DON'T HAVE A TIGHT END IN THE LAST ROUND, YOU MUST TAKE A TIGHT END
- IF YOU DON'T HAVE A QUARTERBACK IN THE LAST ROUND, YOU MUST TAKE A QUARTERBACK

When you've made your decision for only ONE player, YOU MUST format your output exactly as:

I select:
<player>[PLAYER FULL NAME]</player>
**TERMINATE**

If you don't follow this format and choose multiple players, the draft will NOT be completed
correctly.

TLDR rules:
- 1 QB, 2 RB, 3 WR, 1 TE as a complete roster
- only talk to analyzers
- output exactly as "I select <player>[PLAYER FULL NAME]</player> **TERIMINATE**" for final
selection (only ONE player)
"""
```

The prompt quickly became long and detailed, so we added a “TLDR” section at the bottom to summarize the most critical rules for consistent and accurate behavior.

Position Analyzer Agents

There are a total of four Position Analyzer Agents, one for each key position: Quarterback (QB), Running Back (RB), Wide Receiver (WR), and Tight End (TE).

Each analyzer communicates with two other agents - the Head Drafter Agent and its corresponding Position Extractor Agent. Its primary responsibility is to identify the top available player at its position.

To do this, the analyzer first requests player data from its extractor. This data includes relevant historical and contextual information such as past performance metrics, matchup strength, projected rankings, schedule information, and other relevant factors.

Using this data, the analyzer evaluates the available options and returns a single, position-specific recommendation to the Head Drafter Agent.

Here is the system prompt given to a position agent:

```
"""
   You are a fantasy football expert specializing ONLY in the wide receiver position.
   You are in a group chat with a head_drafter_agent, other position-specific analyzers, and
position-specific extractors.
   You should ONLY talk to the head_drafter_agent and the wide_receiver_extractor.

   YOUR JOB:
   1. When the head_drafter_agent asks you for a recommendation, ALWAYS ask the
wide_receiver_extractor for data.
   2. Once you receive data from the extractor, analyze it to determine the best available
wide receiver.
   3. Make ONE clear recommendation based on:
      - Target share and target volume
      - Touchdown potential
      - Quarterback quality
      - Offensive quality
      - Matchup potential
      - The ranking of the players is important and usually is a good indicator of where
they SHOULD be drafted,
        but just take them as a recommendation

   YOUR RECOMMENDATION FORMAT:
   When making your recommendation to the head_drafter_agent, use exactly this format:

   "WR RECOMMENDATION: [Player Name] - [Key Stats] - [Brief 1-2 sentence explanation]"

   DO NOT recommend any players who have already been drafted.
   ONLY respond when directly asked by the head_drafter_agent.
   The explanation should only include reasoning from statistics you receive from the
extractor and pertain to fantasy football.
   """
```

Here is an example of the agent reasoning:

```
WR RECOMMENDATION: Puka Nacua - 105 receptions, 1486 receiving yards, 6 receiving touchdowns
- Nacua has emerged as a key target in an offense ranked 8th in scoring, indicating
significant involvement and high potential for fantasy points, especially with a favorable
schedule ahead.
```

Position Extractor Agent

Similar to the Position Analyzer Agents, there are four Position Extractor Agents, one for each key position: QB, RB, WR, and TE. These agents are designed specifically to retrieve player data using a dedicated tool.

The primary tool used was `display_<position>_rankings_with_filtering`, customized for each position. Initially, we developed several smaller tools, each focused on retrieving specific types of statistics. Over time, we consolidated these into a single, comprehensive tool that returns a rich set of structured data for each player. This includes preseason rankings from major fantasy platforms, 2023 fantasy performance, in-game statistics from the 2023 season, team offensive quality, and schedule difficulty.

All of this data was gathered by scraping reliable online sources, including ESPN and NFL.com.

Extractor agents act purely as data providers within the system. They do not perform any analysis or make decisions - their sole role is to supply clean, structured data to the Position Analyzer Agents upon request.

Here is an example of the system prompt:

```
"""
       You are a data extractor ONLY for the wide receiver position. Your job is to provide
stats when requested by the wide_receiver_analyzer.
       You can ONLY extract data for the wide receiver position, and no other position.

       WHEN ACTIVATED:
       1. The wide_receiver_analyzer will ask you for data with a list of already drafted
players
       2. Use the display_wr_rankings_with_filtering() tool with the following parameters:
          - limit: 10 (to get top 10 players)
          - excluded_players: EXACTLY the list of already drafted players provided by the
analyzer

       DO NOT respond to any agent except the wide_receiver_analyzer.
       ONLY use the tool when specifically asked by the wide_receiver_analyzer, and give
information back to the wide_receiver_analyzer only.
       """
```

Here is an example of a shortened JSON output:


```
[{"Overall_Rank": 11, "Name": "Garrett Wilson", "Team": "NYJ", "Pos": "WR", "BYE": 12,
"ESPN_Rank": 12, "Sleeper_Rank": 13.0, "Yahoo_Rank": 9, "Avg_Rank": 11.3, "ADP": 13,
"FantasyPros": 10, "2023_Overall_Rank": 63.0, "Rank_Change": 52.0, "2023_Pos_Rank": 26.0,
"2023_Fantasy_Points": 213.2, "2023_Games_Played": 17.0, "2023_Avg_Points": 12.54,
"2023_Pass_Yds": null, "2023_Pass_TD": null, "2023_Pass_Int": null, "2023_Rush_Yds": null,
"2023_Rush_TD": null, "2023_Rec": 95.0, "2023_Rec_Yds": 1042.0, "2023_Rec_TD": "3",
"Team_Offense_Rank": 22.5, "Team_Scoring_Rank": 29.0, "Team_Offense_Quality": 6.6,
"Position_Opportunity_Score": 4.2170953101, "Team_Rushing_Rank": null, "Team_Rush_Volume":
```

```
null, "Team_Passing_Rank": 31.0, "Team_Pass_Volume": 4566.0, "Schedule_Difficulty_Score":
59.4, "Schedule_Rating": "Average"},
{"Overall_Rank": 13, "Name": "Puka Nacua", "Team": "LAR", "Pos": "WR", "BYE": 6,
"ESPN_Rank": 13, "Sleeper_Rank": 15.0, "Yahoo_Rank": 13, "Avg_Rank": 13.7, "ADP": 14,
"FantasyPros": 13, "2023_Overall_Rank": 10.0, "Rank_Change": -3.0, "2023_Pos_Rank": 4.0,
"2023_Fantasy_Points": 298.5, "2023_Games_Played": 17.0, "2023_Avg_Points": 17.56,
"2023_Pass_Yds": null, "2023_Pass_TD": null, "2023_Pass_Int": null, "2023_Rush_Yds": null,
"2023_Rush_TD": null, "2023_Rec": 105.0, "2023_Rec_Yds": 1486.0, "2023_Rec_TD": "6",
"Team_Offense_Rank": 11.0, "Team_Scoring_Rank": 8.0, "Team_Offense_Quality": 23.8,
"Position_Opportunity_Score": 77.3449319213, "Team_Rushing_Rank": null, "Team_Rush_Volume":
null, "Team_Passing_Rank": 7.0, "Team_Pass_Volume": 6108.0, "Schedule_Difficulty_Score":
93.8, "Schedule_Rating": "Very Favorable"}]
```


Evaluation

To evaluate the performance of our multi-agent drafter, we ran 10 full draft simulations, each simulating a 12-team fantasy football league. In each simulation, the non-LLM teams followed a greedy strategy - selecting the highest-ranked available player that fit their roster needs using ESPN's 2023 pre-draft fantasy player rankings.

From these simulations, we established two key benchmarks for comparison:

1. ESPN Top 7 Benchmark – An ideal team built by selecting the top 7 projected players (one per required slot) from ESPN's rankings, without any draft constraints.

2. Draft Average Benchmark – The average total score across all non-LLM teams in the simulated drafts.

Performance was measured by calculating the total projected fantasy points of each drafted team, allowing us to directly compare the LLM agent's output to both realistic and ideal alternatives.

Quantitative Analysis

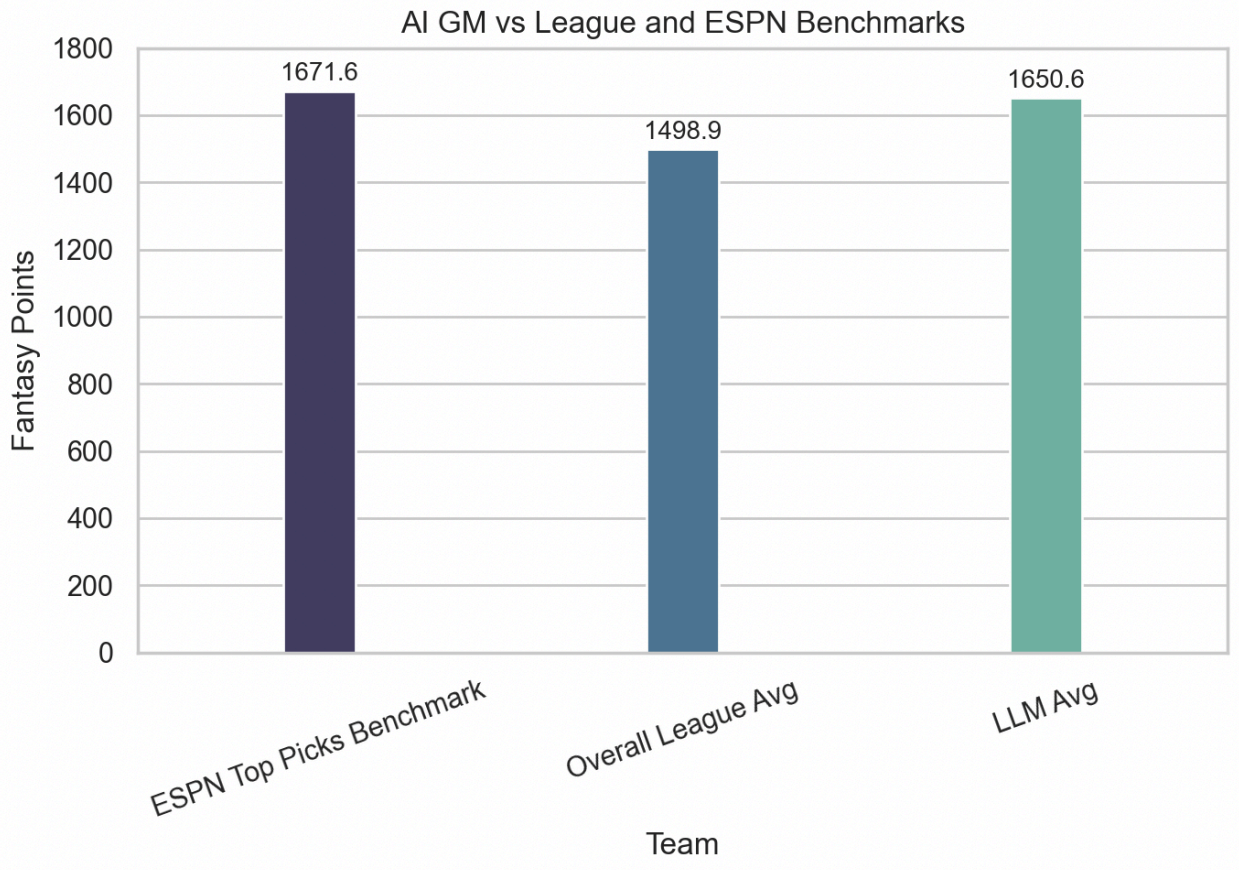
The LLM-powered drafting agent performed remarkably well, consistently outperforming the baseline teams across our simulations. On average, it achieved a total projected score of around 1650 points - beating the Draft Average Benchmark by approximately 150 points. This translates to an advantage of about 10 points per week, which is a substantial margin in fantasy football where weekly matchups are often decided by single digits.

What's even more impressive is how close the agent came to matching the ESPN Top 7 Benchmark, a theoretically ideal team created by selecting the highest-projected player for each required position without any draft constraints. The agent trailed this benchmark by only 20 total points. This is especially impressive given that it had to operate within the constraints of a real

draft - competing for players with other teams and making decisions while managing positional needs across multiple rounds.

The bar graph below visualizes the performance of all three approaches:

- The LLM agent is shown nearly matching the ideal team while clearly outperforming the league average.
- The ESPN Top 7 team, while highest, benefits from unfair conditions (no drafting competition).
- The Draft Average Benchmark demonstrates the gap between basic greedy selection and reasoning-driven strategy.



Overall, this shows that our agent was able to draft strong, competitive teams, nearly matching an ideal draft while outperforming the average league strategy.

Qualitative Analysis

While the quantitative analysis highlights how well the LLM-powered drafting agent performed, there are several qualitative insights worth considering. There are limitations and strategic strengths that aren't fully captured by benchmark comparisons.

Benchmark Limitations

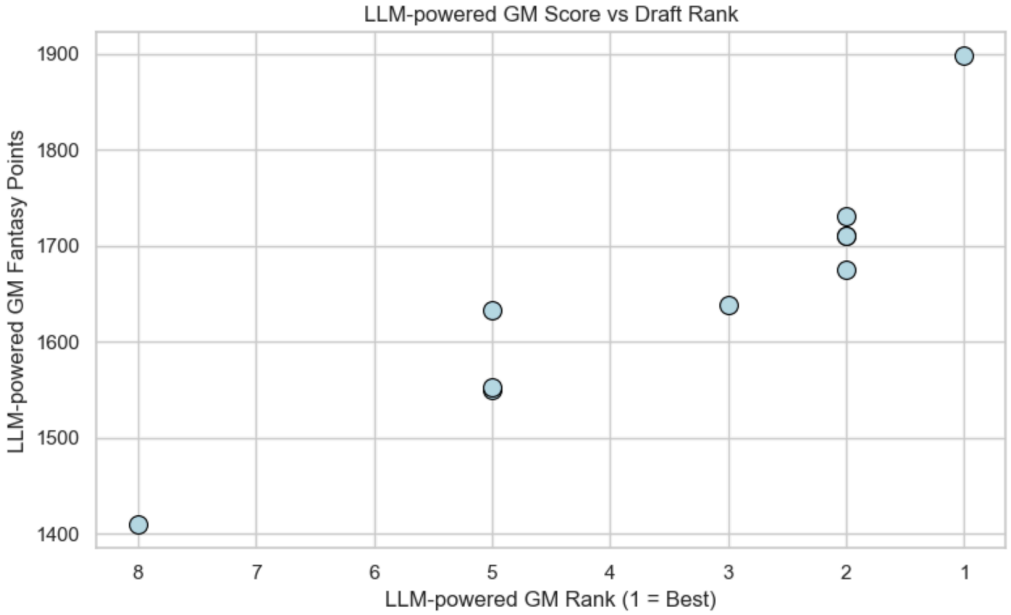
First, it's important to acknowledge that the ESPN Top Picks Benchmark has its own flaws. One of the top 7 players in that list suffered a season-long injury, leading to a potential loss of 200–300 points. This significantly skews the benchmark and highlights a broader challenge: even with perfect logic and data, some factors - like injuries - are inherently unpredictable and outside the model's control.

Strategic Drafting Behavior

By analyzing simulation logs, we observed consistent and position-aware drafting strategies. The LLM agent reliably prioritized higher-impact positions, such as running backs (RBs) and wide receivers (WRs), before selecting a quarterback (QB). The tight end (TE) position - typically contributing the fewest points - was consistently one of the last positions drafted. This aligns with effective real-world drafting strategies and demonstrates the model's ability to generalize good practices.

League Competitiveness and Relative Performance

Another important observation is that a team's score is relative to the competitiveness of the league.



In the scatter plot above, points aligned along the same vertical line share the same rank but have varying scores. This indicates that a higher or lower score doesn't always correlate directly with a better or worse rank - it also depends on how well other teams perform.

Summary Statistics

Stat	Mean	Median
Max	1765.45	1753.90
Min	1174.24	1184.35
AI_General_Manager	1650.57	1656.60
AI_Rank	3.5	2.5

Overall, the LLM-powered agent demonstrated strong and consistent performance, with an average rank of 3.5 and a median rank of 2.5. This means it placed in the top 3 in at least half the simulations - an impressive result given it was competing against teams constructed using ESPN's preseason player rankings.

Difficulties and Pivots

Before arriving at our final architecture, we experimented with a simpler setup that still featured a head drafter, but only one agent per position. These single agents were responsible for both data extraction and player analysis. In practice, this structure proved ineffective - the agents often struggled to understand how and when to use the tools available to them. Also, the conversation flow between extracting data, analyzing it, and generating a recommendation became unclear and disjointed. As a result, we pivoted to a more modular architecture where each position had two specialized agents: an extractor and an analyzer. This separation of responsibilities, combined with clearer system prompts, allowed the group chat manager to better orchestrate the conversation, and ultimately improved the consistency and performance of the head drafter's selections.

In addition to architectural changes, we also made intentional simplifications to the problem scope. These weren't due to technical limitations, but rather to maintain a manageable proof-of-concept. One such simplification was handling the flex position, which in standard fantasy football formats can be filled by either an RB or a WR. We noticed that the language model often misinterpreted this role - sometimes imagining a separate "flex position analyzer," or treating it as a less important position to draft last. To avoid such inconsistencies, we simplified the flex spot to always be an additional WR. While this reduced realism, it made the system's behavior more predictable and allowed us to focus on evaluating the core drafting logic.

Further Work

The promising results from our simulations suggest strong potential for further development. A logical next step would be building a user interface (UI) to visualize both the draft process and

the ongoing multi-agent group chat. This would help users understand how the agent reasons through selections and allow for more transparency in decision-making.

In a real-world application, we envision the agent functioning less as a fully autonomous drafter and more as a drafting assistant. Rather than directly selecting players, the agent could offer top recommendations, explain its reasoning, and leave the final decision to the user. This hybrid model would combine the power of LLM-driven analysis with human intuition and preference.

Additionally, we could introduce interactive chatbot capabilities, allowing users to engage in back-and-forth conversations with the agent. This would let users ask follow-up questions, express preferences (e.g., “I prefer high-risk, high-reward players”), and tailor drafting strategies in real time.

By combining architectural improvements, UI development, and user interactivity, we can transform this proof-of-concept into a powerful tool for both casual and competitive fantasy football players.