EDH-LLM: Using LLMs to build MtG Decks

Theo Hui tchui@ucsd.edu
Linus Lin l6lin@ucsd.edu
Evelyn Yee eyee@ucsd.edu
Jingbo Shang jshang@ucsd.edu

1 Broad Problem Statement

The goal of this project is to generate Magic: The Gathering decks for the commander format (also known as EDH; Elder Dragon Highlander) as a demonstration of the strength of machine learning paradigms built on top of Large Language Models (LLMs) such as ChatGPT. The EDH format presents a loosely structured format for deck-building; players must select a Commander card (see Figure 1a for an example) to lead 99 other cards in a 1v1v1v1 last-man-standing format. Building a successful deck necessitates the following considerations with varying levels of subjectivity:

1. Hard Restrictions:
    - The commander must be a Legendary Creature (more on keywords/typing later). There are over 900 Legendary Creature cards.
    - The other 99 cards in the deck must fit into the color typing of the commander.
2. Soft Restrictions:
    - Synergies: Cards in a deck must play well with each other. This can be expressed in many ways, including typal synergies, playstyle archetypes, and combos. This presents a challenge as there are over 27,000 unique magic cards at the time of writing, and each card may present multiple possible avenues for synergy.
    - Power Curve: Decks must strike a balance between three main archetypes of cards; card draw, mana generation, and gameplan cards must be evenly balanced in a deck to allow a smooth gameplay experience. Missing a turn due to lack of cards, mana, or failing to execute a gameplan are all risks that can be minimized by good deck composition.
    - Rule Zero: Decks must be fun to play *with* **and** play *against*. This can be achieved by building a deck that is, on average, of comparable speed/strength to other decks in the playgroup.

To address these diverse constraints, we frame the challenge of deck building as an information retrieval task, selecting the remaining cards for a commander like a search engine selects relevant documents for a query. Inspired by this framework, we propose a pipeline for leveraging numerical card data and card body text for synergy-informed deck building. We feature a Large Language Model (LLM) in our pipeline in order to address the subjective, holistic demands of this deck building task and get the maximum utility from the semi-structured text data.

2 Narrow Problem Statement

2.1 Theoretical Motivation

The sheer quantity of possible cards available for deck building causes the process of manually creating unique decks to become incredibly daunting. Naively searching through 27,000 cards to find 99 to suit your deck is impractical, and the evaluation of these sub-

jective qualities is difficult to automate using plain heuristics. We are inspired by these challenges to research a way to efficiently generate legal decks that also meet the various soft restrictions of the EDH deckbuilding problem.

With our data consisting primarily of text, natural language processing (NLP) tools, such as Word2Vec (Mikolov et al. 2013) and ChatGPT, can help us find similarities between documents and significantly reduce the search space. Word2Vec transforms text into its vector representations, which will help us find similarities between documents, acting as a broad search for related cards. ChatGPT has significant potential to help us narrow down the documents even further, as it has been trained on several hundred gigabytes of text with billions of parameters, allowing us to implicitly access large embeddings with strategic prompting. ChatGPT will allow us to semi-automatically cluster documents together through the use of guided prompts. This can either be done through the use of iterative prompts, asking ChatGPT to pick which of two documents is most similar to a cluster (triplet task), or simply iteratively prompting ChatGPT to the add single most similar document to the cluster.

2.2 Application – Magic: The Gathering

Our novel approach in our pipeline is the feeding of the body text (oracle text) of the card to our model to identify synergies. The oracle text of MtG cards often contains nuanced and intricate rules, interactions, and conditions. A large language model, such as ChatGPT, can effectively parse and interpret this textual data, recognizing note just specific keywords and mechanics, but contextual information that define the cards' functionalities. The model's extensive training dataset allows it to grasp the subtleties of MtG card language, enabling it to discern not only explicit card effects but also implicit interactions that may lead to synergies.

Card Anatomy:

- Card Name: Located at the top of the card, this name is unique for each card.
- Mana Cost: Positioned in the upper right corner, the mana cost represents the resources required to cast the spell. Colored and/or colorless mana symbols indicate the types and quantities of mana needed.
- Card Type: Beneath the mana cost, the card type specifies whether the card is a creature, instant, sorcery, enchantment, artifact, planeswalker, or land. This classification determines how the card behaves during gameplay. Many cards reference typing for effects and can lead to tribal themes (ex. all elves)
- Oracle Text: The rules text, located below the flavor text (if present), provides a precise description of the card's abilities and effects. Oracle text clarifies any changes or updates to the card's functionality from its original printing.

(a) Sample Commander

(b) Extremely synergistic card

Figure 1: These cards share a high synergy despite having relatively different (tokenwise) oracle text, requiring a higher understanding

2.3 Related Work

EDHRec is the current most popular deck analysis tool, generating card recommendations by scraping human-made deck data from deck build/publishing sites. It is able to recommend cards that are highly synergistic with a chosen commander via an algorithm similar to TF-IDF. This provides insight on cards that interact well with a specific commander vs. commander staples that are generically good. However, this site has major drawbacks:

- Many commanders can utilize multiple theme archetypes (ex. +1/+1 counters, tribal, token, aristocrat, blink). For a given commander, EDHRec will recommend the best cards out of each viable theme, however including all of these cards in a deck would result in an overall weak deck due to a lack of synergy within the other 99.
- Many commanders are used in "Pre-con" decks sold ready-built by Wizards of the Coast. Due to this, on deck sites, most decks including these commanders will have little/no modification. This gives the illusion that many weak cards are in fact synergistic due to their inclusion in this commander's decks.
- EDHRec is incapable of finding new synergies that have not already been found. For example, when new cards are released, EDHRec is unable to provide the same level of recommendations as more established commanders for weeks.

From a more academic perspective, current research regarding deckbuilding systems is also not perfectly suited for this application. For example, Zhang et al. (2022); Kowalski and Miernik (2020) focus on small-scale deckbuilding (i.e. selecting cards from a set

of 5 option), and Fancher (2015); Stiegler et al. (2016) implement heuristic-based search algrithms for assembling decks based on complex characteristics. This effort, though effective at test time, requires significant human supervision/labeling for training, and does not allow for the discovery of new synergies. In contrast, (Kowalski and Miernik 2020; Chen et al. 2018) use sequential processes, like the evolutionary algorithm, to assemble decks, which can be computationally intensive and slow at test-time. Historical systems also rely heavily on play data and do not leverage the information provided in the semi-structured oracle text of the card itself. We re-frame this process of deck-building as a text-based information retrieval task, where a user provides a seed card (i.e. their Commander), and the rest of the deck is built by searching through the text representations of the remaining ~27,000 cards for the most relevant supporting cards.

Information retrieval (IR) tasks are typically separated into two phases: using a fast method to select a mid-sized candidate pool from the large available search space, and then using a slower, more powerful neural system to re-rank this candidate pool for more fine-grained relevance. We aim to explore the use of Large Language Models (LLMs) for the re-ranking aspect of this task, taking advantage of these systems' ability to understand complex relationships in text. This application of LLMs for search tasks has been surveyed by Lin, Nogueira and Yates (2021), pho provide a comprehensive view of applications of large transformer-based language models for single-stage and multi-stage text ranking for a variety of IR purposes. However, these methods are computationally intensive, as they require specialized training of millions of parameters over a large text corpus. To counter this need for custom LLM training, Sun et al. (2023) propose a pipeline for text ranking using large black-box LLM systems, like ChatGPT. They find significant performance improvements over other LLM approaches, including a passage re-ranking BERT model (Nogueira and Cho 2020) and a custom LLaMa model (Touvron et al. 2023). Inspired by these findings, we hope to apply a similar ChatGPT-based approach to re-ranking in our specialized deckbuilding task.

3 Primary Output

3.1 Methods

For our deck-building tool, we present a 2-phase information-retrieval pipeline, using the Commander card as the search query. In the candidate pool selection phase, we embed each card into a high-dimensional vector representation and select the top 500 most similar cards according to cosine similarity. To generate each card's vector representation, we pass its oracle text through an open-sourced text-based embedding model, like the classic Word2Vec (Mikolov et al. 2013) or the more advanced E5 system (Wang et al. 2022) (if we have time and enough computational resources to run E5). We augment these text embeddings with other domain-specific, non-text features of the cards, like card type and color.

After reducing the full list of ~27,000 available cards down to a candidate pool of 500, we will use ChatGPT, through the OpenAI API, to break ties and select the best 99 cards to

form a competitive deck.

3.2 Evaluation

From the over 900 potential Commander cards, we will select a set of 100 for validation/pipeline tuning and 200 for testing our final system. For each Commander, we will evaluate our generated deck against 2 baseline decks:

- **Greedy Popularity Heuristic**: Select the 99 cards which are most frequently played with the selected Commander, according to decklist data from EDHRec (See Section 2.3).
- **Embedding-only**: Gather the 99 cards with the most similar vector embeddings to the commander. Similar to directly sampling a candidate pool of size 99.

We will evaluate each deck (our decks, and the baselines) on the following criteria, related to the constraints described in Section 1:

Hard Restrictions: We will automatically assess the color typing of the cards in the deck, to see that they match the commander, which is a hard restriction of decks in this play format.

Synergy Heuristics: We will use scraped deck data from EDHRec to estimate the synergy between a pair of cards through a Bayesian probability measure. We define the synergy heuristic for two cards a, b as follows:

$$\text{synergy}(a, b) = \frac{P(A \cap B)}{P(A)P(B)}$$

where A and B represent the event of a deck containing cards a and b, respectively, and P denotes the probability of these events over a random deck in the EDHRec corpus.

If cards have negative/no synergy, they should co-occur less frequently together than they do on their own ($P(A \cap B) < P(A)P(B)$). If they have neutral synergy, their occurrence should be independent (i.e. $P(A \cap B) = P(A)P(B)$). If they have positive synergy, they should co-occur more frequently than they do separately ($(P(A \cap B) > P(A)P(B)$)

We will report the **average synergy** over all pairs in a deck (4,950 pairs in a 100-card deck) as well as the **Commander synergy** between all non-commander cards and the commander (99 synergy scores).

Playability Evaluation: To assess the subjective aspects of deckbuilding, we will develop a deck evaluation questionnaire to rank the proposed deck on a variety of qualities, like power balance, win speed and fun-factor, aggregating the scores for each question to get a holistic playability score for the deck. We will prompt ChatGPT to evaluate each deck according to these questions. We will also have humans perform the same playability evaluation on a few decks, to estimate the bias/accuracy of the GPT evaluations. Due to the cost and level of domain expertise required for these human evaluations, we will probably only perform this human evaluation for a few of the 600 decks (ours + 2 baselines, for 200 Commanders), focusing on the decks generated by our system.

3.3 Deliverables

We will deliver a machine learning tool to generate effective, fun, synergistic decks for the EDH format, given the name of a commander as input. Each deck will consist of 99 cards, all of which are among the color typings as permitted by the commander. First, this tool gathers a narrow list of the most similar cards to the commander through the use of vector representations. Next, these cards will be further narrowed down by iteratively prompting ChatGPT to select the cards best suited for the deck. Lastly, we will evaluate the resulting decklist with custom evaluation metrics, such as synergy score and a subjective competitive ranking.

We will also develop a poster presentation and an interactive website to showcase the results of our experiments. On the website, you will be able to choose from a pre-set list of commanders and view the step-by-step results of our experimental processes, as well as the final resulting decklist.

References

Chen, Zhengxing, Chris Amato, Truong-Huy Nguyen, Seth Cooper, Yizhou Sun, and Magy Seif El-Nasr. 2018. "Q-DeckRec: A Fast Deck Recommendation System for Collectible Card Games."

Fancher, Will. 2015. , Sep. [Link]

Kowalski, Jakub, and Radosław Miernik. 2020. "Evolutionary Approach to Collectible Card Game Arena Deckbuilding using Active Genes."

Lin, Jimmy, Rodrigo Nogueira, and Andrew Yates. 2021. "Pretrained Transformers for Text Ranking: BERT and Beyond."

Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. "Efficient Estimation of Word Representations in Vector Space."

Nogueira, Rodrigo, and Kyunghyun Cho. 2020. "Passage Re-ranking with BERT."

Stiegler, Andreas, Claudius Messerschmidt, Johannes Maucher, and Keshav Dahal. 2016. "Hearthstone deck-construction with a utility system." In *2016 10th International Conference on Software, Knowledge, Information Management Applications (SKIMA)*. [Link]

Sun, Weiwei, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. 2023. "Is ChatGPT Good at Search? Investigating Large Language Models as Re-Ranking Agents."

Touvron, Hugo, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. "LLaMA: Open and Efficient Foundation Language Models."

Wang, Liang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang,

Rangan Majumder, and Furu Wei. 2022. “Text Embeddings by Weakly-Supervised Contrastive Pre-training.”

Zhang, Yulun, Matthew C. Fontaine, Amy K. Hoover, and Stefanos Nikolaidis. 2022. “Deep Surrogate Assisted MAP-Elites for Automated Hearthstone Deckbuilding.”