When presented with the same environment and the same amount of resources, an individual’s actions can provide insight into their unique decision-making tendencies and patterns. Modeling cognitive differences between individuals — their behavior, intent, and ultimate goals — allows us to preemptively detect patterns that may foretell future actions. If it is possible to model and predict these unique patterns, such a tool can be tremendously useful for cognitive modeling, simulation, computer security, and various other areas. AI is already at the forefront of research in threat detection (Ahmed and Echi 2021, Lee et al. 2019), but systems that are capable of both identifying entities of interest from their actions and predicting their future actions are still both a highly important and an underdeveloped area of global relations research.

Collectible card games (CCGs) are a type of card game that combines strategic deck building and player-vs-player combat with features of trading cards. Each card has some inherent value relative to the other cards within the same game and contains effects that can be used to engage in combat. CCGs are a practical test space for resource allocation and battle strategy due to their structures which promote reactionary responses to attacks and allow players to obtain more combat power by expending more resources. However, their broad action spaces also allow many possible card combinations and creative multi-turn operations that allow us to model individual cognitive differences instead of abstract generalizations based on the averaged behavior of many players.

However, the existing literature is limited in delineating individual behavior from combined group data in CCGs. Most existing models are capable of recognizing pre-specified card combinations that are common to many players but cannot handle arbitrary sequences of cards that may be unique to individuals. In this work, we aim to provide a starting point for using transformer-based language models to identify individualized features in CCG play styles.

This paper will address two specific research questions

RQ1 Given a player’s last move in a particular game state, can we identify the player?

RQ2 Given a player in a particular game state, can we predict their next action?

We make two major contributions in the addressing of these questions: 1. We introduce a dedicated and expansive dataset which captures the entire game state of a virtual CCG and links that state to a unique player and their action, resulting in nearly two million tuples of gameplay data. 2. We train three transformer-based large language models — DistilBERT (Sanh et al. 2020), RoBERTaBase, and RoBERTaLarge (Liu et al. 2019) — on our dataset to predict certain player features. We also train a variety of simpler classification models on our dataset for comparison.

To study the extent to which transformer-based language models can answer our research questions, we use Legends of Code and Magic (LoCM), an online CCG where two players (or programmers) create bots to proceed through the drafting and battling. In the drafting phase, both players are presented with the same three cards, and each must select one to place in their deck. This process continues iteratively until each player has selected 30 cards. In the battle phase, each player is randomly dealt cards from their deck to their hand and “summons” cards from their hand to play. Cards can be creatures or items; creatures are played against an opponent to either deplete the opponent’s health or the health of their creature cards. Item cards can increase a player’s health, deplete the opponent’s health, or add or remove abilities from creatures already in play. The game ends when a player has no health remaining.

LoCM has the noted advantage of being specifically designed for AI research and testing (Kowalski and Miernik 2020). It is also playable entirely online, thereby reducing some of the technical overhead required for an AI to play. However, LoCM has a significantly smaller action space, with only 160 cards with deterministic effects, compared to the over 4000 with more complex effects in other CCGs such as Hearthstone. Further, LoCM has an established player base, so any model to identify individual players or predict their actions can have a robust validation set. Finally, using bots to engage in battles fixes each player’s style in a game, allowing for clear delineation between styles and strategies.

To create a dataset with several unique players, we scraped the LoCM leaderboard for the top thousand players, each identified by a unique gamer identification number. We then used the CodinGame API to obtain the most recent games played by each player. Games were removed if the opponent was the LoCM “Battlemage” AI or if the game
ended with fewer than 64 turns, indicating that a player encountered a coding error during or immediately following the drafting phase. The resulting games were then parsed to extract each player’s starting hand, the subsequent cards they drew, and their actions. From this information, we could reconstruct the game state at each turn in the game. Each entry in the resulting dataset corresponds with one turn and contains the player ID for the player engaging in that turn, the game ID, the cards in the player’s hand, the cards on the board, identified by whether they belong to the player or their opponent, and the player’s action.

This produced nearly two million turns for 1,319 unique players over 30,299 games. The dataset is highly unbalanced to favor a few especially active players, which creates the potential for a model to overpredict those players to arbitrarily improve accuracy. As such, for testing purposes, we reduced the dataset to only include players who had between 1000 and 2000 overall turns. The resulting testing set contains 228,487 turns for 184 unique players over 6,367 games.

The dataset was then split into train, validation, and test sets by randomly selecting 80% of the game IDs for the train set and 10% each for the validation and test sets. The split was done to include entire games within a singular set, but split individual players across sets, so that the model could not predict a player’s actions based exclusively on the game, as the state may remain relatively fixed for large periods of time within a game. The game ID was then removed from all splits as it was no longer needed.

To identify a player from the game state and action and determine which features of the game state were most relevant to player identification, we set the player ID as the gold label. Similarly, we set the action taken as the gold label when predicting an action from the game state and player. We then trained on three separate feature sets augmented with the non-gold label:

- **State**: The entire train dataset consisting of the player’s hand, their board, and the opponent’s board;
- **Hand**: A reduced train dataset consisting only of the player’s hand;
- **Board**: A reduced dataset consisting of the player’s board and the opponent’s board.

In order to evaluate whether our trained model was able to perform better than arbitrary models, we also created multiple baselines:

- **Random**: This model randomly selected a player from the entire space of players in the train set. However, as a part of the train data, the model is presented with a set of cards that exist in the game state at each turn. The resulting action must be bounded by these available cards. As such, at each turn, we constructed a set of possible moves from the cards available in the game state, and this model randomly predicted an action from this reduced action space.

- **Majority Class**: To examine the possibility that the models were simply selecting the most common option in order to arbitrarily increase their accuracy, this model selected the label with the most occurrences in the entire space.

- **No State**: To examine the possibility that the models were simply learning to identify a player based on the cards they drafted, or the most common moves a specific player, made without consideration for the state, this model selected a player using only the player’s hand — which is randomly drawn from the drafted deck — and an action using only the player ID, with no other information about the game state.

The input to every model was the feature set being tested — with each feature labeled — conjoined into a single string.

To create a baseline for comparing our transformer-based approach, we first trained five simple classifier models: Naïve Bayes, XGBoost, decision trees, random forest, and a linear support vector classifier (SVC). We limited the decision trees and random forest classifiers to a maximum depth of 14 in order to balance speed with performance. For these baseline classifiers, the input string was vectorized using TF-IDF vectorization. We then trained three transformer-based models, each obtained from Huggingface, on each task: RoBERTa-base, RoBERTa-large, and DistilBERT-base-uncased.

For each transformer-based model, the input string was tokenized using the Huggingface tokenizer specific to that model. They were then finetuned for up to 30 epochs, but we stopped training when three consecutive epochs showed no improvement in F1 score. Every model had a learning rate of $5 \times 10^{-6}$, a batch size of 16, a maximum sequence length of 512, and used FP16 half precision training for speed. We used the model with the best F1 score on the validation set to perform predictions on the test set.

The best overall model was RoBERTa_large presented with the State feature set, with 19.60% accuracy and a 17.38% F1 when identifying the player and 31.07% accuracy and a 1.71% F1 when predicting an action, outperforming both DistilBERT and RoBERTa_base. Additionally, all three Transformer models outperformed the random baseline, indicating that they were actually learning about the input data instead of simply guessing, and the majority class baseline, indicating that they are capable of identifying unique features about all of the players.

RoBERTa_large also outperformed all of the non-transformer classifiers except in the No State feature set, suggesting that more complex computation is needed to differentiate between player features. Although DistilBERT and RoBERTa_base underperformed XGBoost and LinearSVC in many feature sets for identifying a player, both models performed on par with or above all of the non-transformer classifiers when presented with the entire game state and outperformed all non-transformer classifiers in predicting the action.

RoBERTa generally outperforms DistilBERT on all feature sets, but the difference becomes less significant as the number of game state features decreases. Eventually, the difference becomes statistically negligible for the No State feature set. This may indicate an upper bound on the amount of information that can be obtained from the cards that a player drafts, without considering actions. It may also indicate the impact that the game state has on player decisions.
References


