Transformer-Based Approaches to CCG Design Aids:
Using BERT and T5 for Card Authentication and Generation

Henry Lane

Abstract

Long-lived collectible card games face increasing challenges as they age to produce new and refreshing content that nonetheless remains faithful to game flavor and identity, which is a core draw for many players. Language models may present a new avenue to develop tools to assist the design process to provide gut checks of faithfulness as well as inspiring new mechanics. This project explores the use of transformer-based language models to develop two tools for the games *Hearthstone* and *Android: NetRunner* that might be used to these ends.

Introduction

Collectible Card Games (CCGs), Trading Card Games (TCGs), and Living Card Games (LCGs) are a family of games that feature a large collection of unique playing cards, which players collect and use to construct decks of cards to then duel one another with. Perhaps the most famous of these is *Magic, The Gathering* [1], which was created in 1993, but it and other well-known entries like *Pokemon* [2] and *Yu-Gi-Oh* [3] have long-lasting appeal, with competitive and casual scenes thriving nearly three decades later. Part of the enduring appeal is the availability of distinct classes, or factions, or tribes or cards within these games that have unique feels and patterns of play, again allowing for player expression. These distinctions are often subtle and difficult to articulate beyond “you know it when you see it.” A *Magic, The Gathering* player, for instance, might be a die-hard “Red Deck” player as it offers them a tailored experience within the rules of the game that fits how they like to play.

Maintaining this “flavor” within the game is important to retaining each game’s identity and unique appeal among its playerbase. Designers need to make sure new releases contain fresh ideas, while ensuring that new products are faithful to the original vision. This task is made more difficult by the fact that these factions are largely defined by unwritten rules. It can be difficult to judge the degree to which any given card “feels” like it belongs to its assigned faction; conversely it can be challenging to think of ideas for new cards that fit within the faction. This is the motivating challenge behind this research project.

In this project, I explore ways of using machine learning and natural language modeling techniques to develop faction-agnostic design aids. Specifically, I explore a model that can accurately classify the faction of a given card given a text representation of it, and a model that is capable of generating novel cards when prompted with a specific faction. The former could be useful for assessing whether or not a new idea for a card is faithful to the faction’s flavor within the existing card pool, and the latter could be useful for generating initial ideas for new cards within a faction when designers face writer’s block. This project uses data on cards from two popular CCGs: *Hearthstone* [4] and *Android: Netrunner* [5], which have ~5,000 and ~3,000 unique collectible cards respectively. While prior work has been done in the realm of card generation, to my knowledge this work is novel in its focus on faction classification, and the first to consider *NetRunner.*

Below are examples of cards from both games, cards 1 and 2 from *Hearthstone* and 3 and 4 from *NetRunner*, from left to right. “Sleepy Dragon” is a “Neutral” faction card, whereas “Cenarius” is a “Druid” faction card. Both are relatively high-costed minions that include the keyword “Taunt”, but the Druid card also has a “Choose One” effect, and theming that is more distinctly “Druid”. For the *NetRunner* cards, we have two 1-cost assets with a trash cost of three. However, “B-1001” is “Neutral” and has an effect that is relevant to all factions, whereas “Wall to Wall” is the “Weyland Consortium” faction and has an effect that includes advancing ICE (another type of card), which is a distinctly “Weyland” effect. The Neutral cards in both cases also tend to be simpler than their faction specific counterparts.

Related Work

Prior approaches to modeling CCGs with machine learning have used both RNN and transformer architectures, and have experimented with creating new sets of embeddings as well as leveraging pre-trained transformers.

Janusz et al. [6] take the idea that embeddings of the text of cards can contain useful predictive information to predict card interchangeability for *Hearthstone* cards. They employ a neural network that leverages word2vec embeddings for the text of cards and predict, for a card in a given deck, the probability of other cards being included in that same deck. Janusz et al. found that the inclusion of textual information from the card and the wiki articles for those cards yield more accurate predictions than using the numerical stats of the card alone.

Zuin et al. [7, 8] use output from a LSTM model with self-attention coupled with a gradient boosted tree to create a classifier that takes components of *Magic, The Gathering* cards (e.g. faction, card text, rarity), to attempt to predict the “Mana Cost” of the card, which is a proxy for the power level of the card. While my focus is on the faction of a card rather than the cost, it is similar to the work of Zuin et al. insofar as both are classification problems that take advantage of embedded meaning in the card text to improve predictions. Indeed, Zuin et al. found that including textual representations yielded far superior accuracy than models using the numerical components of the card alone.

Summerville et al. [9] focus on generating novel card text for *Magic, The Gathering* cards, making use of sequence to sequence generation with attention to improve upon prior attempts that used pure LSTM recurrent neural networks. Summerville et al. apply a denoising encoder-decoder LSTM with an attentional system applied to the decoder when reading the outputs of the encoder. They train the model to de-noise incomplete or garbled existing cards and return the true card text. When presented with various pieces of an incomplete custom card, the model can then attempt to fill in the missing pieces.

Proposed Approach

This project uses transformers for both the classification task and the generation task. For the classification task, I make use of pre-trained BERT [10] and RoBERTa [11] models which I then fine-tune for the classification task, and use the CLS token output with a categorical classification head to predict the factions of cards. This model is evaluated against a baseline of two other approaches for classification in the form of decision trees, and a traditional neural network. Decision trees and neural networks are both able to learn about the standalone implications of and interactions between various components of the cards, such as the cost, stats, keywords, etc. While this can be useful information, alone it's not typically sufficient to predict the faction of a card[1]. In order for the card game to be balanced across classes in the first place, it necessarily must be the case that stats and keywords alone aren't sufficient to determine class; "Hunters" having only cheap minions and "Mages" having only expensive ones would make balancing the game difficult. The key advantage of a transformer like BERT is the ability to take all of the other game text on the card that decision trees and neural nets are unable to make meaningful use of, and learn things about it[2]. By fine-tuning a BERT model on examples of *Hearthstone* cards, this project aims to capture the subtle semantic differences in how cards of various factions are worded, and the flavor differences in how they are named. RoBERTa is also explored to assess whether or not the different pre-training used by RoBERTa produces a model more suited to classification of cards.

For the generation task, this project explores two approaches to fine-tuning the T5 [12] transformer model to produce novel cards. T5 is chosen for its encoder-decoder architecture, and because it was trained on several prefix based tasks including question and answering. This project takes advantage of this by formatting the generation prompt in a way that resembles a question and answer, with a new prefix for generation, and fine-tuning T5 on this new task. In addition and unlike prior papers that looked at card generation, this allows me to inject "noise" into the prompt in the form of a randomly generated seed[3]. The goal is for the additional source of noise to reduce the need to take samples over large numbers of possible words when performing generation.[4] This project also examines a secondary architecture wherein the fine-tuned generation model is fed into a generative adversarial network comprised of the generator, and a BERT-based discriminator with a classification head that predicts if a card it sees is real, or a generated fake. The hope is that this secondary pass of training allows the model to learn to produce more realistic seeming cards, after it learns the general sense of what a card should look like.

Data

This project uses cards from *Hearthstone* and *Android: NetRunner*. Cards were collected from the Firestone application for *Hearthstone* [13] and from the NetRunnerDB database for

---

[1] For instance, with *Hearthstone* a 3-cost, 3-health, 4-attack, "beast" type minion with the keyword "rush" provides lots of information, but only the type (beast) actually narrows the faction down, and even then this could easily be a "Hunter" card or a "Druid" card, as both factions often have "beast" type minions at their disposal.

[2] Consider again our example of the *Hearthstone* card, and now suppose the title is "Oakgrove Defender" and the card text says "Whenever your hero attacks without a weapon equipped, it gains +2 attack." This is very distinctly "Druid" in terms of naming flavor and the way the card functions, and information that would be inaccessible to the other types of models.

[3] For example "generate: A Hunter, Minion card using seed dizzy ladder"

[4] Default settings result in the model regurgitating the same card often despite the random noise, so I also take advantage of sampling over some K mostly likely words when generating, which produces a good level of card novelty.

Android: NetRunner [14]. Both games share some fundamental aspects when it comes to the anatomy of a card, which this project uses to model card features. A detailed breakdown is included in the appendix. Prior to being used within the models evaluated, the key features of each card (e.g. cost to play, health, cost to trash, etc) as well as any explicitly mentioned game mechanics were extracted into individual features, so that complete text descriptions of each card could be constructed. Below are two examples of the constructed descriptions, one from *Hearthstone* and one from *Android: NetRunner*.

(left, *Hearthstone*): The card named Cenarius is a 8 cost minion with 8 health and 5 attack, and includes the effects choose one. The card text says: Choose One plus Give your other minions plus 2 dash plus 2; or Summon two 2 dash 2 Treants with Taunt.

(right, *Android: NetRunner*): The card named B-1001 is a 1 cost asset with the subtypes enforcer, bioroid. It has an influence requirement of 0. It has a trash cost of 3. The card text says Remove 1 tag: End the run. You cannot use this ability during a run on this server.

Hearthstone features cards from 12 factions: Paladin, Hunter, Shaman, Warrior, Druid, Demonhunter, Deathknight, Warlock, Priest, Mage, Rogue, and Neutral. As a result of a large printing of neutral cards in each set and a later introduction of the Deathknight and Demonhunter classes in the games lifestyle, there is an imbalance in the data in terms of representation. In order to address this, half of the neutral cards were randomly omitted and each card from the other classes was included twice, to obtain a more balanced dataset.

Android: NetRunner is slightly more complicated in that it features two “super factions” in Corporation and Runner, that contain 4 and 3 distinct factions respectively – and there are also 3 runner “mini-factions” with an extremely limited number of examples of cards (12 each). For this dataset, the mini-factions were omitted leaving the remaining, largely balanced dataset containing the following factions: Jinteki, Haas-Bioroid, NBN, Weyland, Neutral Corp, for the corporations, and Criminal, Anarch, Shaper, and Neutral Runner for the runners.

Evaluation

The first task addressed by this project is classification of the correct factions of cards. In this case the chosen metric to optimize is the model accuracy, since specific precision and recall score would have to be computed separately for each faction. The generation task requires a more subjective approach to evaluating the quality of generated cards, since there are no correct answers for what a new card might be. This project considers both a “Legibility”[5] score and a “Faction Faithfulness”[6] score for the generated cards. In addition to these subjective measures, which I tag manually for a subset of the generated cards, I compare model

[5] Scored from 0 to 3 with the following criteria: 0) mostly or completely incoherent text; 1) approaching coherence but with hallucinations or made up rules; 2) mostly coherent, but over or under powered; 3) could exist as a real card.

[6] Scored either 0 or 1 with the following criteria: 0) does not fit the flavor of the assigned faction; 1) does fit the flavor of the assigned faction.

performance using two objective metrics: ROUGE scores, which are primarily for scoring summarizations but provide some additional insight into the faithfulness of the design for the quality of the generations compared to reference texts; and BLEU scores, which despite being originally for machine translation tasks still provides an alternative measure of textual similarity. This project does not consider model perplexity as Summerville et al. [9] use for measuring their *Magic: The Gathering* card generation performance; so long as the generations remain coherent, a higher perplexity is not inherently bad as part of the use of such a model in the real world would be novel card concepts.

Results

Below are the compiled results of the models evaluated, beginning with the classification task. Each cell contains the performance on both the *Hearthstone* cards in blue, and the *NetRunner* cards in purple. The most performant model is **bolded**.

Model	**Accuracy**	**Precision**	**Recall**
Decision Tree	0.35 / 0.49	0.47 / 0.48	0.35 / 0.49
Gradient Boosted Tree	0.41 / 0.50	0.47 / 0.49	0.41 / 0.50
Hyperparam Tuned Neural Net	0.36 / 0.49	0.44 / 0.48	0.36 / 0.49
RoBERTa Classifier	0.73 / 0.59	0.74 / 0.63	0.73 / 0.59
BERT Classifier	**0.77 / 0.68**	**0.78 / 0.69**	**0.77 / 0.68**

Of the model specifications tried, the transformer-based models are far superior to the specifications that were unable to make use of the text on the cards, with BERT being best overall. Evidence that the additional information contained in the card name and game text is useful to the model is borne out by the confusion matrices and an examination of the top attention layer (see appendix for images). The model can sometimes identify key words within the card text (i.e. “Demon” as clear markers of a given class to make a prediction; conversely, the model is more likely to get confused by instances of these words appearing on cards of classes they don’t usually appear for.[7][8] The model also has more trouble with “Neutral” cards than faction specific cards, which makes sense as the absence of extra flavor and typically less complicated design can make them harder to pin down. Similarly with the *NetRunner* cards, the model rarely gets confused about the super faction (“Runner” or “Corporation”), but does make some errors between factions of the same type. One other interesting finding is the model is better than expected at discerning “Neutral” cards. I hypothesize this is because “Neutral” Netrunner cards tend to have an influence requirement of 0, whereas other factions always have an influence of 1 or more. The model might then be “cheating” by using this feature of the card as a clear signal that it’s neutral.

[7] For instance, for the NetRunner model the words “suffer” (usually some kind of damage) is typically an Anarch effect; this made the model mistake a Shaper card that would “prevent an instance where you would suffer damage” as an Anarch card (see appendix).

[8] Interestingly for *Hearthstone*, the model tends to predict incorrectly more often for “adjacent factions” (see appendix). For example, for “Warlock” cards when the model was wrong it most often predicted “Priest”, an adjacent faction.

The next table presents the metrics from the evaluation task for both models that were experimented with: fine-tuned T5, and fine-tuned T5 plus GAN training. ROUGE-2 and ROUGE-L are preferred as word ordering matters more for coherence of card texts.

Model	ROUGE-2	ROUGE-L	BLEU	Legibility	Faction Faithfulness
Fine-tuned T5	0.428 / 0.452	0.587 / 0.575	0.465 / 0.491	1.87 / 0.81	0.35 / 0.19
Fine-tuned T5 + GAN	0.432 / 0.455	0.588 / 0.578	0.469 / 0.497	— / —	— / —

The results here suggest that the fine-tuning approach of using a GAN architecture was either ineffective, or too resource intensive to be possible using the resources available while conducting this project. The model training loop was quite slow and memory limits hindered the possible batch sizes; the model failed to make meaningful training improvements in the epochs I was able to run it for. Of additional interest is that while these metrics would suggest similar performance on the *NetRunner* and *Hearthstone* cards, subjectively the cards generated by the *Hearthstone* model were much more realistic and coherent than their *NetRunner* counterparts.

Due to the similarity of performance and time constraints of the project, only the batch of generations from the original T5 model were assessed via subjective scoring. On average, the *Hearthstone* cards were close to being real cards if a little over or under-tuned, while the vast majority of *NetRunner* cards were mostly or completely incoherent and rule-breaking, with frequent hallucinations of made up game mechanics. Further, about 1 in 3 *Hearthstone* cards “fit” with the faction they were generated for, while only 1 in 5 did for *NetRunner*. This is likely due to the difference in underlying complexity of the two games. The average *Hearthstone* card text contains 11.3 words after processing and requires 34.4 words to describe fully, whereas the average *NetRunner* card text contains 27.5 words and takes 55.2 words to describe fully. *NetRunner* cards also feature a larger diversity of effects, many of which should never appear outside of a specific subtype of card. The appendix features some examples of better and worse cards. The model is capable of producing cards that approximate the types of words that should appear on a netrunner card, but often on the wrong type of card or in the wrong order. Future work to train models on subsets of the data for runner and corp cards separately might produce better results.

Conclusion and Future Work

This project demonstrates the promise of using language-based machine learning models as assistive design tools in the CCG development space. A model capable of learning the “read-between-the-lines” interpretation of what a specific faction in a card game means can be a helpful second opinion for designers looking to make sure they are staying true to the core flavor of a game. While the generative models in this project proved unable to generate game-legal cards, they did manage to produce some interesting ideas that could help as a creative spark for designers. Future work in this space could focus on models such as GPT-2 or LLAMA-2 to assess whether a larger model, expressly designed as a decoder only, could produce better results.

Appendices

Detailed Breakdown of Card Anatomy

- **Cost:** In both games, there is a cost to playing/using a given card; this number is in the upper left-hand corner of the card. While cards from all factions can have different costs, there are some factions (particularly in *Hearthstone*) that tend to have lower costs.
- **Other numbers:** Both games also have some other key features; in *Hearthstone* there is the damage/attack and health/armor value of many cards, and in *Android: NetRunner* there are numbers for strength, cost to trash, and influence for deck building requirements.
- **Card types/subtypes:** Both games have different types of cards (spells, minions in *Hearthstone*, programs, assets in *NetRunner*), with some factions seeing more or less of a given type than others. There are also unique subtypes in both games (e.g., “beast” or “mech” in *Hearthstone*, or “Bioroid” or “Grey Ops” in *NetRunner*), which also leave some additional context for what faction a card might come from.
- **Special keywords:** *Hearthstone* has a number of game mechanics that appear as keywords on specific cards (e.g. “charge”, or “taunt”), which are tagged in the input data and also appear more/less frequently for certain factions. While no exact analogue exists for *Android: NetRunner*, there are nonetheless interactions with certain mechanics that are more specific to some factions than others (e.g. “tags” or “sabotage”).
- **Card titles and text:** Last, each card also has other text on it in the form of the card title itself, and the accompanying rules text that helps determine how the card should work. Again, this often contains subtle information that tends to be unique for how each faction in both games play.

Figure A1. Confusion Matrix for *Hearthstone* BERT Classification

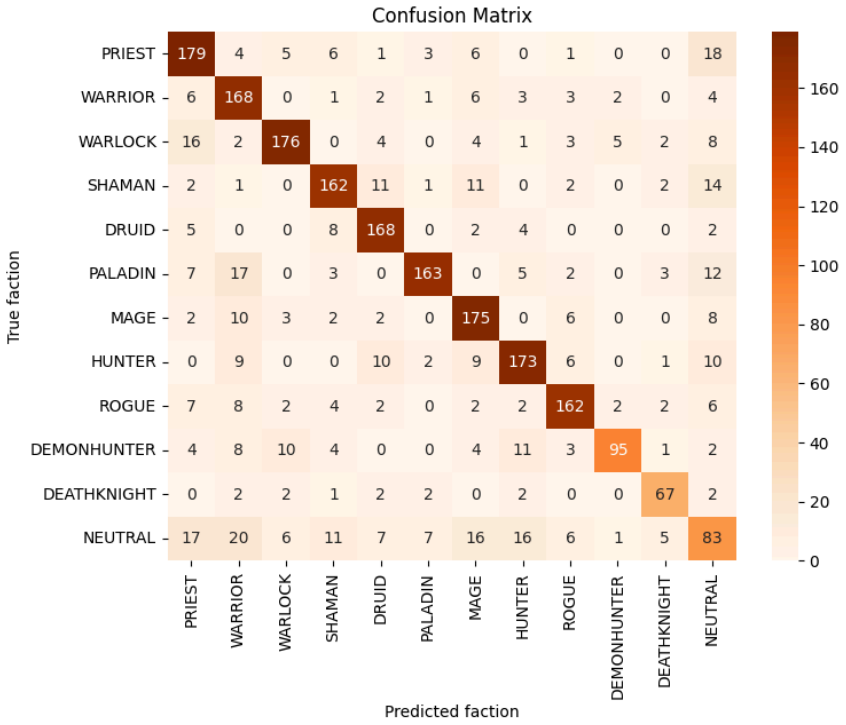


Figure A2. Confusion Matrix for *NetRunner* BERT Classification

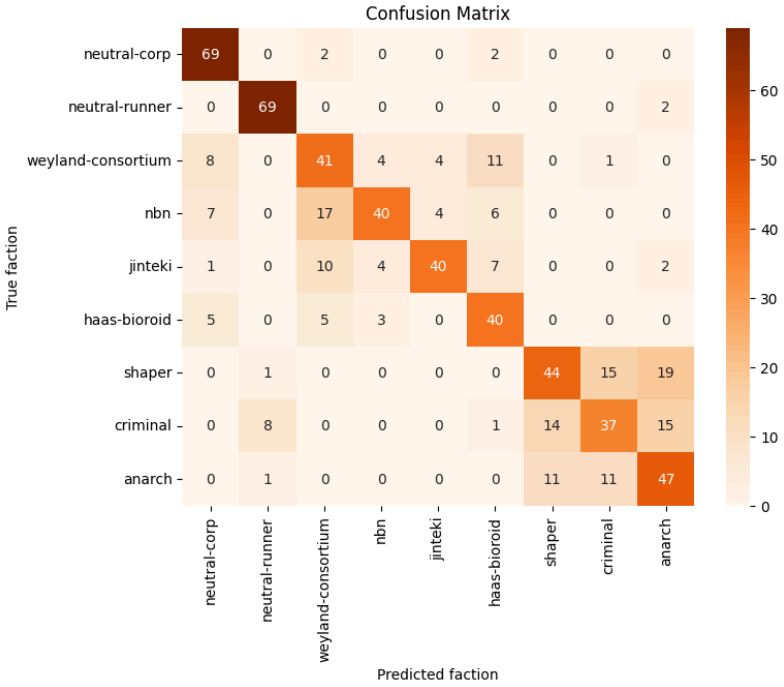
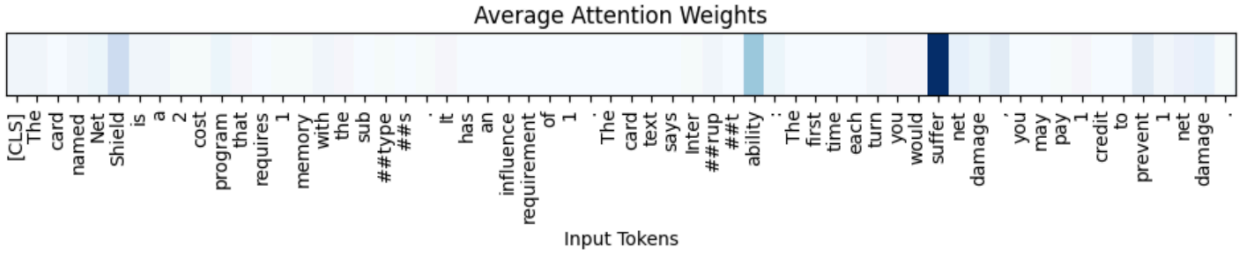
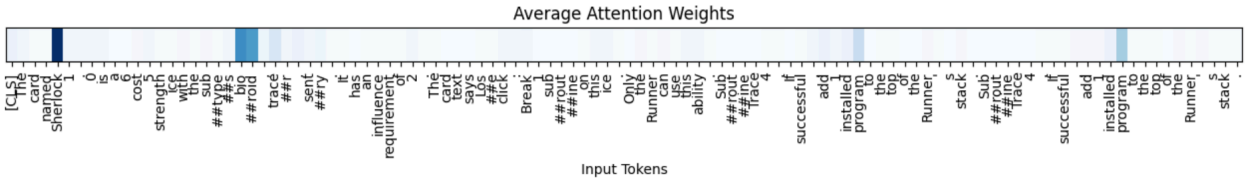


Figure A3. Sample Attentions from NetRunner and Hearthstone Models

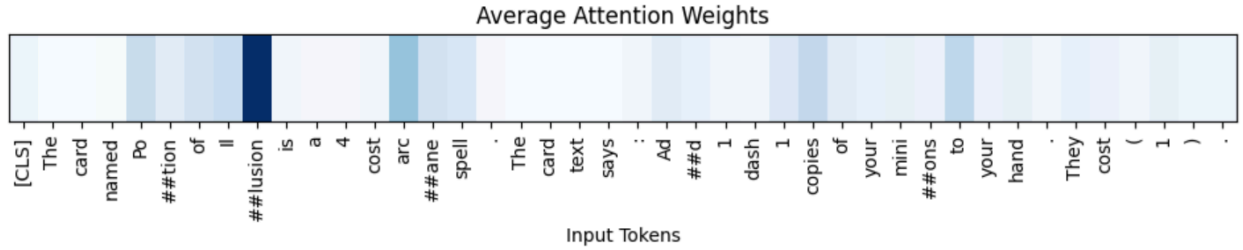
Incorrectly predicted as “Anarch” when actually “Shaper”



Correctly predicted as “Haas-Bioroid”



Incorrectly predicted as “Mage”, actually “Rogue”



Correctly predicted as “Warlock”

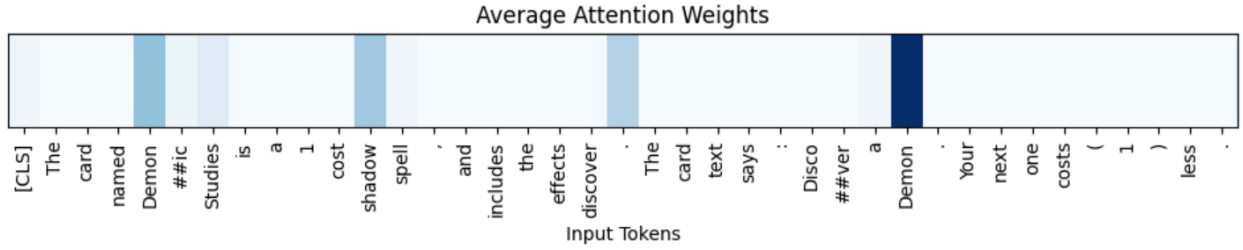


Figure A4. *Hearthstone* Class wheel

Note: There is now technically an 11th class with the addition of “Deathknight” to the game, but the amount of cards for this class are smaller currently due to the recency of the addition.

Example Generated Cards

Hearthstone

- [Priest Minion, scored 3 for legibility and 1 for faithfulness]: The card named Ossilar Baronet is a 6 cost dragon minion with 5 health and 6 attack, and includes the effects deal damage, battlecry. The card text says: Battlecry: Deal 3 damage to a random enemy minion.
- [Hunter Minion, scored 2 for legibility and 0 for faithfulness]: The card named Cobra Blade is a 7 cost beast minion with 4 health and 3 attack, and includes the effects battlecry. The card text says: Battlecry: Shuffle 2 Shadow spells into your deck. If two are drawn each turn, you gain plus 3 Attack.
- [Warlock Minion, scored 1 for legibility and 1 for faithfulness]: The card named Elffayan Shadowslinger is a 4 cost undead minion with 5 health and 4 attack, and includes the effects deathrattle, rush, trigger visual. The card text says: Rush. At the end of your turn, the next Level you play that

NetRunner

- [Neutral Corp Agenda, scored 3 for legibility and 1 for faithfulness]: The card named Optical Exchange is a 3 advancement agenda worth 1 points with the subtypes security. It has an influence requirement of 0. The card text says Click: Draw 2 cards.
- [Jinteki Agenda, scored 2 for legibility and 0 for faithfulness]: The card named "Cosmic Transformation" is a 23 advancement agenda worth 3 points with the subtypes security. It has an influence requirement of 2. It has a trash cost of 4. The card text says When you score this agenda, gain 4 credits.
- [HB Agenda, scored 1 for legibility and 0 for faithfulness]: The card named Research Prototype is a 5 advancement agenda worth 2 points with the subtypes research. It has an influence requirement of 0. The card text says While research is ongoing, the Corp gains 0 credits.
- [Weyland Asset, scored 0 for legibility and 0 for faithfulness]: The card named Sniper is a 1 cost asset with the subtypes cyber. It has an influence requirement of 1. It has a trash cost of 1. The card text says Whenever you play the first item of clothing or other card from headquarters to headquarters, you may trash 1 card from headquarters to headquarters as an asset. Add Sniper to the end of the run.

References

1. R. Garfield, *Magic: The Gathering*, [CCG]. Wizards of the Coast, Renton, WA, USA, 1993.
2. *Pokemon*, [TCG]. Media Factory, Japan, 1996.
3. *Yu-Gi-Oh*, [CCG]. Konami, Japan, 1999.
4. B. Brode, *Hearthstone,* [CCG]. Blizzard Entertainment, Irvine, CA, USA, 2014.
5. R. Garfield, *Android: NetRunner*, [LCG]. Fantasy Flight Games, Roseville, MN, USA, 2012.
6. A. Janusz, Ł. Grad and D. Ślęzak, "Utilizing Hybrid Information Sources to Learn Representations of Cards in Collectible Card Video Games," 2018 IEEE International Conference on Data Mining Workshops (ICDMW), Singapore, 2018, pp. 422-429, doi: 10.1109/ICDMW.2018.00069.
    a. https://ieeexplore.ieee.org/abstract/document/8637417
7. G. Zuin and A. Veloso, "Learning a Resource Scale for Collectible Card Games," 2019 IEEE Conference on Games (CoG), London, UK, 2019, pp. 1-8, doi: 10.1109/CIG.2019.8847946.
    a. https://ieeexplore.ieee.org/document/8847946
8. G. Zuin, L. Chaimowicz and A. Veloso, "Deep Learning Techniques for Explainable Resource Scales in Collectible Card Games," in IEEE Transactions on Games, vol. 14, no. 1, pp. 46-55, March 2022, doi: 10.1109/TG.2020.3030742.
    a. https://ieeexplore.ieee.org/document/9222105
9. Summerville, A., & Mateas, M. (2021). Mystical Tutor: A Magic: The Gathering Design Assistant via Denoising Sequence-to-Sequence Learning. Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, 12(1), 86-92. https://doi.org/10.1609/aiide.v12i1.12851
    a. https://ojs.aaai.org/index.php/AIIDE/article/view/12851
10. Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. ArXiv abs/1810.04805 (2018)
    a. https://arxiv.org/abs/1810.04805
11. Liu, Yinhan et al. “RoBERTa: A Robustly Optimized BERT Pretraining Approach.” ArXiv abs/1907.11692 (2019)
    a. https://arxiv.org/abs/1907.11692
12. Raffel, Colin, et al. “Exploring the Limits of Transfer Learning With a Unified Text-to-Text Transformer.” arXiv.org, 23 Oct. 2019, arxiv.org/abs/1910.10683.
    a. https://arxiv.org/abs/1910.10683
13. Firestone - a Hearthstone deck tracker and companion app. (n.d.). https://www.firestoneapp.com/, https://static.firestoneapp.com/data/cards/cards_enUS.gz.json
14. Android: Netrunner Cards and Deckbuilder · NetrunnerDB. (n.d.). https://netrunnerdb.com/, https://netrunnerdb.com/api/2.0/public/cards
15. HuggingFace notebook for Decision Forests
    a. https://www.tensorflow.org/decision_forests/tutorials/beginner_colab

16. Fine-Tuning T5 with Limited Ram
    a. https://github.com/datasci-w266/2024-spring-main/blob/master/materials/walkthrough_notebooks/keras_with_limited_ram/fine_tune_t5_with_limited_ram_pytorch.ipynb
17. Basic Generative Adversarial Network set up in pytorch
    a. https://www.run.ai/guides/deep-learning-for-computer-vision/pytorch-gan#3