Multilabel Classification of Trading Card Game

Matan Diamond
Georgia Institute of Technology
mdiamond8@gatech.edu

Abstract

Trading card games are a unique design space because of the natural categorization that arises from designing similar cards. Magic: the Gathering's color system is a deeply important element of the twenty-eight year old card game. By training a model to classify colors of Magic cards, one can gain insight into the patterns behind the game in a unique way. By using a pretrained transformer model for text embedding, combined with a traditional neural network, I propose a new method of classifying colors of Magic cards. This model opens up a new lens of analysis into the card game.

1. Introduction

A collectible card game, also known as a trading card game, is a card game that involves strategic deck building elements using trading cards, typically acquired through booster packs. The first collectible card game, known as *Magic: The Gathering*, released in 1993 by Wizards of the Coast and was followed by other trading card games such as *Yu-Gi-Oh!* and *Pokémon*.

Magic: The Gathering remains a popular trading card game today, with over 20,000 distinct cards released over the past three decades. Every card has a "mana cost" representing the amount of resources that it costs to use the card. *Magic: The Gathering* uses five colors of mana: white, blue, black, red, and green. This arrangement is known as the "color pie". Devised by Magic creator Richard Garfield, the color system is one of *Magic*'s most fundamental and iconic elements.

Each color of cards in *Magic: The Gathering* is unique mechanically, allowing for differing play styles for decks built with each respective color. Moreover, it is important that "color pie breaks," in which a card is printed with a trait that is not typically accessible through its colors, are rare. This maintains the unique identity and play patterns for each color, which is important for the health of the game.

The goal of this project is to create a classifier that predicts the color(s) of a card, given the card's body. This can be thought of as "learning" what it means for a card to be a color.

1.1. Background

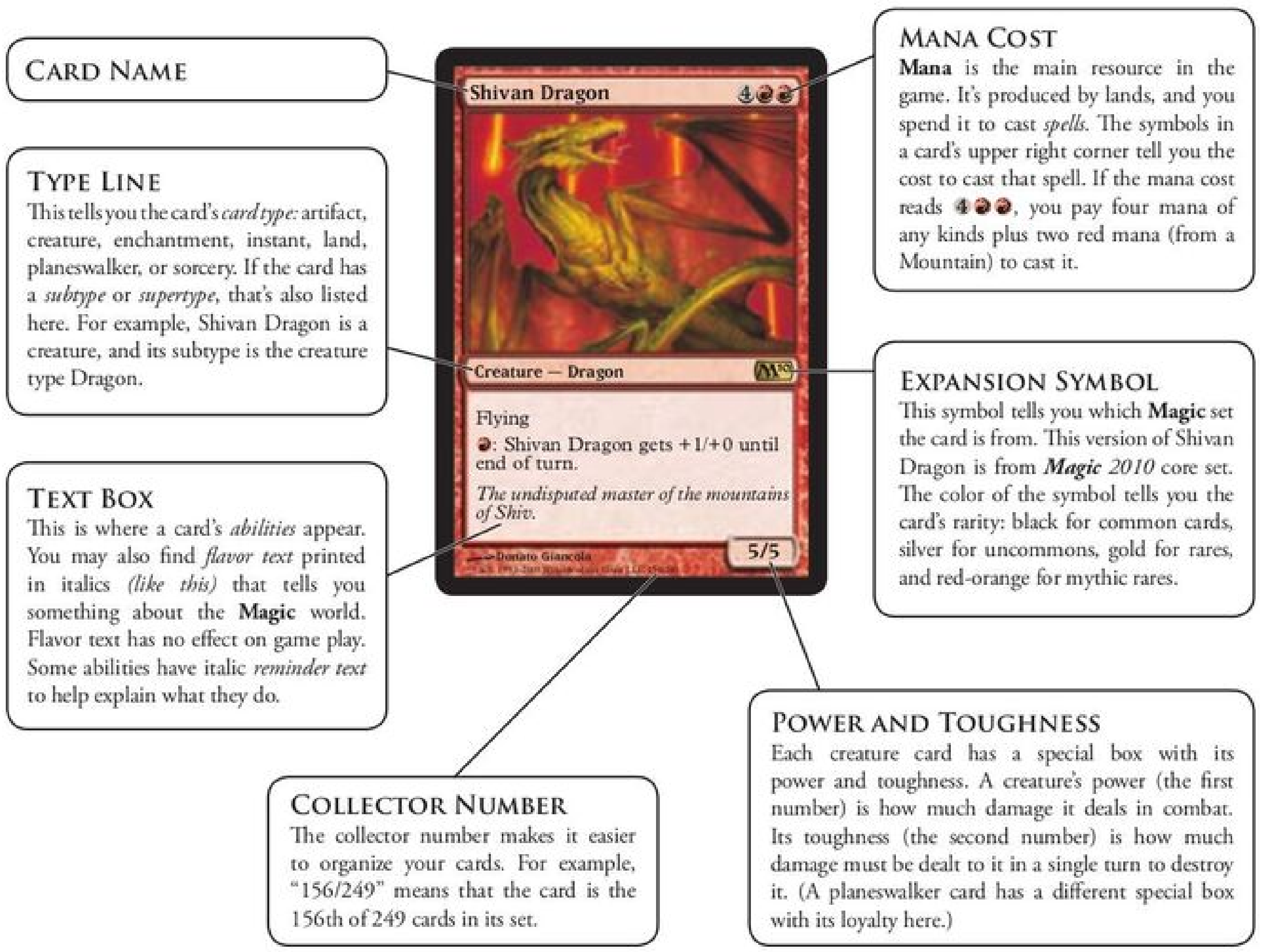
Magic: The Gathering cards are designed by Wizards of the Coast, released in numerous expansion sets every year. The lead designer, Mark Rosewater, occasionally updates the players with articles explaining the mechanical color pie [5]. From this, it is possible to construct specific rules for each color. However, these guidelines are somewhat vague, with primary, secondary and tertiary mechanics attributed to each color. Furthermore, these traits are frequently vague and/or overlap with other colors in nuanced ways. Finally, occasionally a card may contain traits from outside of its color(s) wheelhouse if it does so by fulfilling a mechanic within its wheelhouse. Because of these factors, a strictly rules-based system is in-viable for color classification.

Magic cards follow a consistent structure, as seen in Figure 1. All cards have a mana cost, type line and text box. All *creature* cards also have a power and toughness. While most of these features are either categorical or numerical, the text box is, by definition, text. In order to make use of this text feature, it is necessary to embed the text as a vector so that it can be processed alongside the other features.

Many natural language processing techniques exist for feature-embedding text data. A bag-of-words approach involves recording frequencies of common words in text, and feeding in this histogram as a vector embedding [2]. A recurrent-neural-network such as a long-short-term-memory network (LSTM) is another option for embedding the state of a sentence or paragraph [3]. More recent techniques involve transformers for embedding sentences. I use a well-known transformer architecture called BERT for my text embedding [1].

1.2. Motivation

The *Magic* color pie is a particularly interesting topic from a game design perspective. Arguably, the natures of each color are what make *Magic* compelling as a game. One could potentially argue that a color classifier would have use internally at Wizards of the Coast for designing new *Magic*


Shivan Dragon
4●●
Creature — Dragon
Flying
●: Shivan Dragon gets +1/+0 until end of turn.
The undisputed master of the mountains of Shiv.
5/5
Donato Giancola

CARD NAME

TYPE LINE
This tells you the card's *card type:* artifact, creature, enchantment, instant, land, planeswalker, or sorcery. If the card has a *subtype* or *supertype*, that's also listed here. For example, Shivan Dragon is a creature, and its subtype is the creature type Dragon.

TEXT BOX
This is where a card's *abilities* appear. You may also find *flavor text* printed in italics *(like this)* that tells you something about the **Magic** world. Flavor text has no effect on game play. Some abilities have italic *reminder text* to help explain what they do.

MANA COST
Mana is the main resource in the game. It's produced by lands, and you spend it to cast *spells*. The symbols in a card's upper right corner tell you the cost to cast that spell. If the mana cost reads 4●●, you pay four mana of any kinds plus two red mana (from a Mountain) to cast it.

EXPANSION SYMBOL
This symbol tells you which **Magic** set the card is from. This version of Shivan Dragon is from ***Magic** 2010* core set. The color of the symbol tells you the card's rarity: black for common cards, silver for uncommons, gold for rares, and red-orange for mythic rares.

COLLECTOR NUMBER
The collector number makes it easier to organize your cards. For example, "156/249" means that the card is the 156th of 249 cards in its set.

POWER AND TOUGHNESS
Each creature card has a special box with its power and toughness. A creature's power (the first number) is how much damage it deals in combat. Its toughness (the second number) is how much damage must be dealt to it in a single turn to destroy it. (A planeswalker card has a different special box with its loyalty here.)

Figure 1. The parts of a *Magic: The Gathering* card.

sets. I refute this, and argue that the true value of modeling the color pie is in seeing interactions between the colors, for an analysis of the system as a whole.

For instance, if it turns out that two colors are frequently confused by the model, it indicates that those two colors are too mechanically similar. Thus, some effort can be made to produce more distinct cards of those two colors in the future.

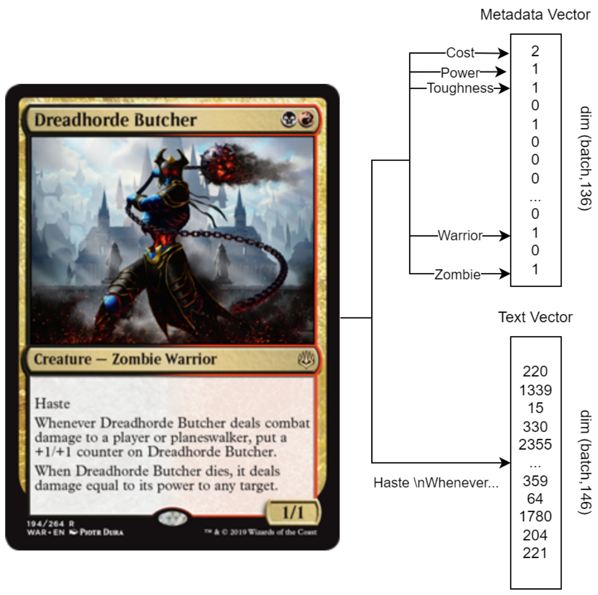
It also serves to show how combinations of colors interact. One of the goals when Wizards of the Coast designs a new set is to determine what identity multi-colored cards will have. Ideally, a blue-green card is mechanically similar to both mono-blue and mono-green cards, but with some notion of distinctness that sets it apart. If a two-colored combination is more frequently confused with one of its base colors over the second base color, then it might indicate that the two-colored identity is skewed too much.

By comparing the modeled color distribution of expansion sets, combined with consumer data of how well those sets are received, Wizards of the Coast could use such a model to inform their designers as to which color pie mechanics are best received by the players.

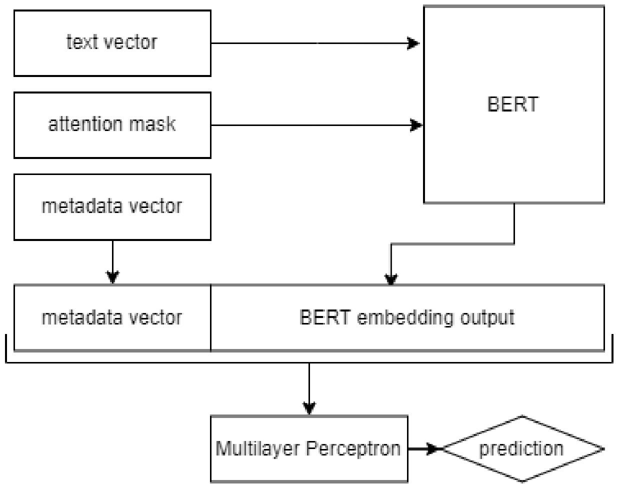
This types of analysis is impossible without a model for the color pie, and by creating this model I can observe the game of *Magic: the Gathering* through a new lens.

2. Approach

I ended up using a deep neural network for the classification task. To accomplish this, I had to clean and reformat a dataset, handle embedding for the textual and categorical card features, and design and train a deep neural network to handle the classification.


Metadata Vector
Cost
Power
Toughness
2
1
1
0
1
0
0
0
...
0
1
0
1
Warrior
Zombie
dim (batch,136)
Text Vector
220
1339
15
330
2355
...
359
64
1780
204
221
Haste \nWhenever...
dim (batch,146)
Dreadhorde Butcher
Creature — Zombie Warrior
Haste
Whenever Dreadhorde Butcher deals combat damage to a player or planeswalker, put a +1/+1 counter on Dreadhorde Butcher.
When Dreadhorde Butcher dies, it deals damage equal to its power to any target.
1/1

(a) An illustration of how cards are converted to tensors.



(b) A diagram of the model architecture.

Figure 2. An embedding example and illustration of the model.

2.1. Dataset

I used a Kaggle dataset containing every *Magic: the Gathering* card since November, 2019 [4]. The dataset contains 20478 unique cards. However, after a data cleaning process, the dataset was trimmed down to 13106 cards.

Cards were removed if:

- They were prohibited from all tournament formats.
- They had no printings from after July 21, 2003 (due to a shift in the color philosophy).
- They were contained in an expansion focused on color pie breaks.
- They were double-faced (they were two cards combined).
- They had a type that is always colorless.

Cards were then split into train and test sets, using a 0.8 train-to-test ratio, resulting in 10724 cards for training and 2382 for testing.

2.2. Embedding

Once filtered, the cards had to be converted to tensors to serve as input for training and evaluation. Two vectors were created for each card, serving as inputs to the model: a metadata vector and a text vector. This process is illustrated in figure 2a.

Numerical features such as the card cost, power and toughness were directly translated to positions in the metadata vector. Then, categorical features such as the type line were encoded as one-hot vectors, each occupying a set amount of space in the metadata vector.

The text box was passed to the BERT encoder, padding to max length, to create the text vector. This process also output an attention mask for each card that was passed alongside the text and metadata vectors to the model.

2.3. Model

The model uses BERT for embedding the text vector. The output of BERT is then concatenated directly with the metadata vector, and the combined vector is input to a multilayer perceptron with one hidden layer separated by ReLU nonlinearity functions. The output is passed through a sigmoid function. The output is 5-dimensional, corresponding to each of the 5 colors. The model architecture is illustrated in figure 2b.

2.4. Training

The model was trained with a batch size of 64, a hidden layer dimension of 128. The model was trained to 100 epochs but it appeared to worsen in performance after 50 epochs, so the checkpoint at 50 epochs was used as the final model. Gradient clipping of 3.0 was used to prevent gradient explosion. Cross entropy loss was chosen as the loss function.

Numerous learning rates were tried, both for the classifier and the BERT parameters. At first, I attempted to train with BERT totally frozen which reached within 1% of the final performance. Ultimately a learning rate of 1e-5 for the classifier and 1e-6 for the transformer proved optimal.

It appeared that higher batch sizes improved training stability, but a batch size of 128 exceeded the GPU memory, so a batch size of 64 was chosen.

Training for 100 epochs took about 5 hours using a GPU on Google Colaboratory, which made grid search too costly to explore every parameter. It is possible that further hyperparameter tuning could improve performance but it appears that the model learned sufficiently as a result of these parameters.

3. Experiments

I used three metrics to measure the performance of the model: accuracy, Hamming loss 1, and Jaccard score 2.

Accuracy is what we ideally want to maximize. However, assessing accuracy alone does not give us a full assessment because it over-penalizes for partially-correct predictions.

The Hamming loss is the fraction of the wrong labels to the total number of labels. This means that guessing red, when the correct answer was blue-red results in only a 0.5 loss. Since this is a loss function, the optimal value is zero with an upper bound of one.

The Jaccard score is Intersection over Union, defined by the number of correctly predicted labels divided by the union of predicted and true labels. Similarly, this gives partial score to half-right choices. This index has an optimal value of one with a lower bound of zero.

The Hamming loss function is defined as:

$$\frac{1}{|N| \cdot |L|} \sum_{i=1}^{|N|} \sum_{j=1}^{|L|} \mathbf{xor}(y_{i,j}, z_{i,j}) \tag{1}$$

Where $y_{i,j}$ is the target, $z_{i,j}$ is the prediction, and $\mathbf{xor}(\cdot)$ is the Exclusive OR operator.

The Jaccard score is defined as:

$$\frac{|T \cap P|}{|T \cup P|} \tag{2}$$

Where P and T are sets of predicted labels and true labels respectively.

4. Results

The resulting evaluation metrics can be found in Table 1. At first glance, the scores may appear sub-par, especially the accuracy result. However, the task of a five-way binary classification means there are 32 combinations possible. Since this is the first model to exist for this specific task, the only comparison is a random guess which would score an accuracy of 3% on average. Thus, I consider this project a remarkable success.

	Accuracy	**Hamming Loss**	**Jaccard Score**
train	0.3237	0.2211	0.3285
test	0.3157	0.2237	0.3273

Table 1. Final Metrics

	acc_train	count_train	acc_test	count_test
types				
Artifact	0.1806	1201	0.1446	235
Creature	0.4311	6049	0.4203	1356
Enchantment	0.1510	1139	0.1352	244
Instant	0.2421	1404	0.2445	319
Planeswalker	0.1172	145	0.1000	40
Sorcery	0.1775	1256	0.1660	289
Tribal	0.3658	41	0.3076	13

Table 2. Accuracy by Type

4.1. Analysis

Moreover, the original goal was to perform analysis based on the distribution of predictions compared to the real labels. Figure 3 shows this through a confusion matrix. On the left hand side is the true color(s) of cards and along each row is the probabilities that the model predicts each other color. This visual was truncated to only contain labels with two or less colors. This is because, while some cards have three, four and five colors, they are quite rare. As a result, the model practically never predicts a card to have more than two colors. This means that (a) cards with three or more colors are almost unilaterally misclassified and (b) cards with two or less colors are almost never classified with more than two colors. Thus, the matrix is truncated for the sake of a better visual tool.

Table 2 shows the accuracy, grouped by the main types of *Magic: the Gathering*. Creatures performed the best, which makes some sense because creatures are the only type that has power and toughness. Tribal performed second best, which is a category with very few cards in it. This is likely because tribal cards are unique in that they can have creature types, which means that tribal cards benefit indirectly from training on creatures (of which there are many more).

Planeswalkers performed the worst, followed by enchantments and artifacts. Planeswalkers are fairly complicated cards with a lot of text, which might have contributed to the confusion whereas enchantments have unique effects, often with very little text.

Artifacts are the one category I feel should have performed better - the greater majority of artifacts are color-

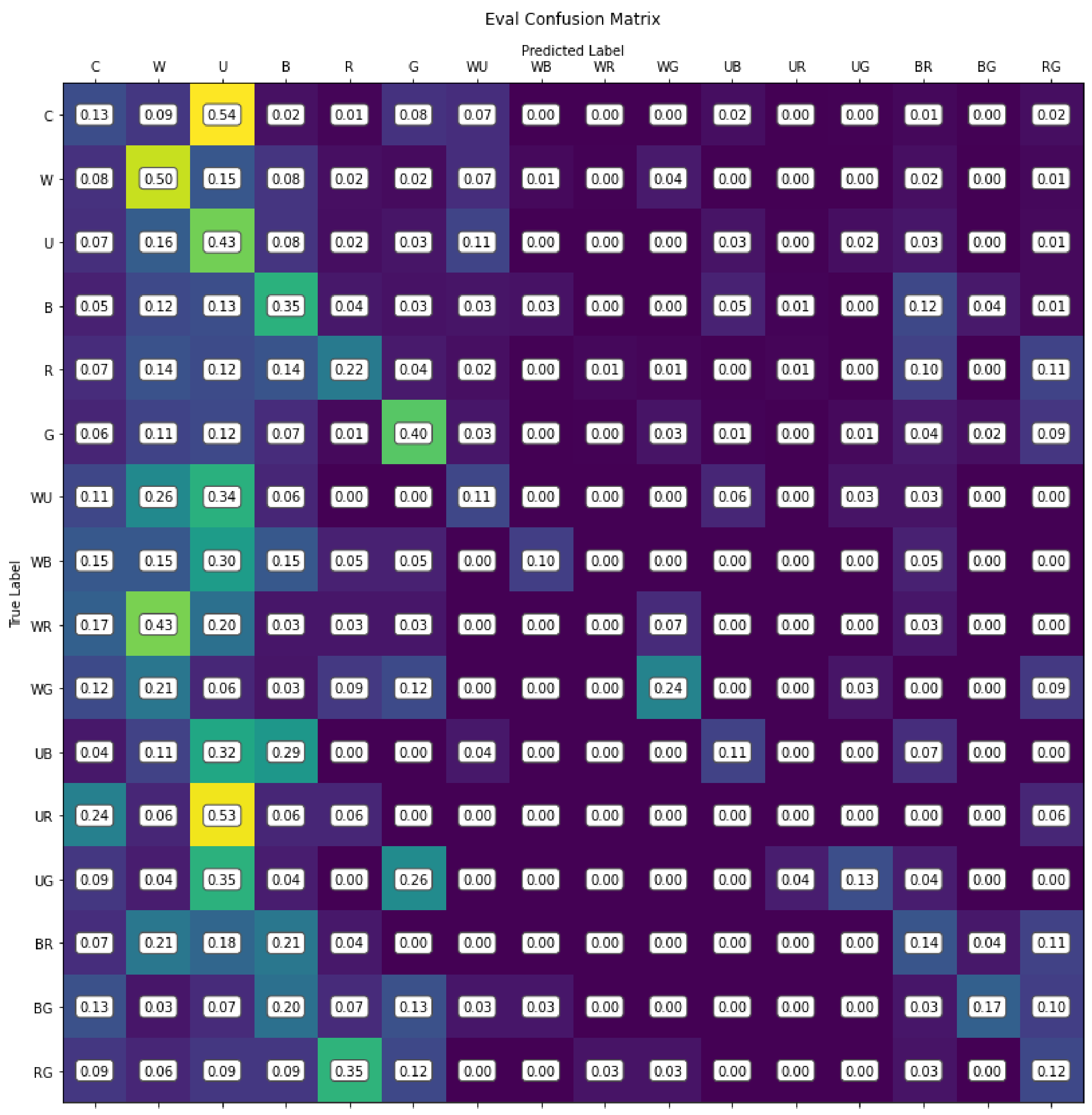


Figure 3. The confusion normalized evaluation data confusion matrix.

less, but perhaps the nature of the five-label output made it difficult for all five color outputs to agree on colorless cards. Furthermore, based on the confusion matrix 3, it appears that most colorless cards were misclassified as blue, which as a color has many artifact synergies. This likely contributed to the low artifact accuracy rate.

Table 3 shows accuracy by subtypes (only showing the highest frequency subtypes). Most of these subtypes are creature subtypes, which we know will perform reasonably well just because creatures as a type performed well on average. Notably aura only appears on enchantments (which performed poorly on average). Equipment only appears on

subtypes	acc_train	count_train	acc_test	count_test
Angel	0.7549	102.0	0.6800	25.0
Aura	0.1379	493.0	0.0795	88.0
Beast	0.4180	177.0	0.3214	56.0
Cat	0.3628	113.0	0.2500	24.0
Cleric	0.6250	200.0	0.6000	50.0
Dragon	0.5112	133.0	0.5238	21.0
Druid	0.7735	106.0	0.7727	22.0
Elemental	0.1311	305.0	0.0571	70.0
Elf	0.7333	240.0	0.7777	63.0
Equipment	0.5277	216.0	0.5135	37.0
Goblin	0.6621	222.0	0.6363	44.0
Human	0.4328	1206.0	0.4346	283.0
Knight	0.5059	168.0	0.5000	52.0
Merfolk	0.5474	137.0	0.5185	27.0
Rogue	0.2435	193.0	0.2647	34.0
Shaman	0.3228	285.0	0.3859	57.0
Soldier	0.6246	349.0	0.5921	76.0
Spirit	0.2721	327.0	0.3015	63.0
Vampire	0.3910	179.0	0.3333	36.0
Warrior	0.4484	504.0	0.3875	129.0
Wizard	0.4825	431.0	0.4811	106.0
Zombie	0.4925	268.0	0.4477	67.0

Table 3. Accuracy by Subtype (Highest Frequency Subtypes)

artifacts and most equipment are colorless (both of which performed poorly on average). However, while auras continued to perform poorly, equipment performed quite well, greatly outperforming the average for colorless cards and artifact cards.

The subtypes which clearly outperformed are primarily those that are devoted to a single color. Specifically, angels, clerics, dragons, druids, elves, goblins and merfolk are all mostly within a single color. It makes sense that these would be easy to predict because there is such a high correlation between the subtype and a single color.

The subtypes which underperformed are the opposite: cats, elementals, rogues, shamans and spirits are all found in multiple colors, so it didn't really help as much to know their subtype for the classifier.

Table 4 shows accuracy by each color. This was grouped as "any card containing that color" which means that, for example, a red-black card is counted as a red card and a black card for the purpose of this aggregation. It appears that white greatly over performed the other colors, which is an indication that white is the most distinct color from the others. Contrarily, red underperformed the most, which indicates the opposite: red is the least distinct from the other colors.

colors	acc_train	count_train	acc_test	count_test
B	0.2910	2230	0.2814	501
G	0.2904	2210	0.3280	500
R	0.2129	2221	0.1780	500
U	0.3571	2212	0.3313	495
W	0.3936	2248	0.3858	508

Table 4. Accuracy by Color (All Cards Containing the Color)

4.2. Future Work

There are ways I have thought of to improve performance that ended up being outside the scope of this project. One such improvement would be to add a few more features. Specifically, adding a card's name, likely fed as characters into an LSTM for text processing, could improve performance. This would especially be relevant for multi-colored cards from specific sets: for instance, in the fictional *Ravnica* setting, two-color combinations have specific names such as "Golgari" indicating green-black or "Izzet" indicating blue-red. The presence of these names could help determine whether a card is mono-colored or two-colored, for instance.

Another interesting feature would be including the expansion set the card is printed in. Similar to above, certain expansion sets are more likely to have two-color or three-color cards, which are difficult to classify right now. One issue with this is that cards often appear to multiple expansions, so finding a way to determine which expansion to tie cards to (or whether to include them all) would have to be determined in the design phase. Along with this, a date of earliest printing could be included for similar reasons. This would also have the upside of potentially learning how Wizards of the Coast's color philosophy has shifted over time.

From an algorithmic perspective, I believe the BERT transformer could have been fine-tuned better. If I were to redo this project, I would fine-tune as a pre-training task on a masked language modelling scenario, which BERT was originally trained on. This might help improve the embeddings, especially for words and symbols specific to *Magic: the Gathering*.

5. Conclusion

This task has been immensely interesting to me, and I believe I have grown a great deal through this undertaking. I have tried to use BERT in the past but was unsuccessful. Thus, I was very happy to get BERT to work for me for this project. Some tricks I learned from this Deep Learning course came in handy - such as gradient clipping, using different learning rates for different parameters in the model, and the general understanding of PyTorch that I now possess.

Furthermore, the task of *Magic: the Gathering* classification was something that I have a lot of excitement for. I actually performed a similar project for my undergraduate Machine Learning course final project a few years ago (but with some important differences). This project has showcased my growth since then, and I am overjoyed at how well the final product worked, and the analysis that it allowed for.

References

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019. 1

[2] Zellig S. Harris. Distributional structure. *¡i¿WORD¡/i¿*, 10(2-3):146–162, 1954. 1

[3] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 11 1997. 1

[4] Mark Rosewater. Magic the gathering cards, 2019. 3

[5] Mark Rosewater. Mechanical color pie 2021, 2021. 1