Title: Card Game Project Writeup

Name: Gabriel Donnan

Number

Contents

[Analysis 3](#_Toc12863164)

[Identification 3](#_Toc12863165)

[Computational Approach 3](#_Toc12863166)

[Research 4](#_Toc12863167)

[Stakeholders 7](#_Toc12863168)

[Interviews 7](#_Toc12863169)

[Hardware Spec 8](#_Toc12863170)

[Software Spec 9](#_Toc12863171)

[Requirements 9](#_Toc12863172)

[Limitations 10](#_Toc12863173)

[Design 11](#_Toc12863174)

[Implementation 12](#_Toc12863175)

[Testing 13](#_Toc12863176)

[Evaluation 14](#_Toc12863177)

# Analysis

## Identification

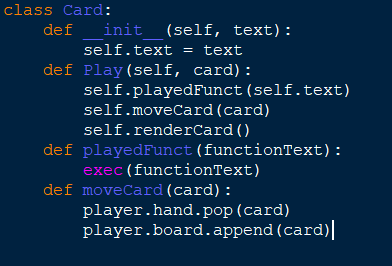
Currently I believe the market for digital card games is relatively bare, with only two main games dominating the market. I believe this leaves room for the creation of a new digital card game, using the simple combat mechanics employed by many card games in the past but using the digital medium to allow for more complicated effects and mechanics that are not amenable to the physical medium, specifically pseudo-random effects such as random generation. As such I believe that creating a game to fill this role would be beneficial to those who may believe the current game market to be stale as it could revitalize an interest in games of this type.

## Computational Approach

The problem of making this is, I believe, best solved by computational methods. This is for a variety of reasons, firstly the game is easily broken down into consecutive logical operations, which is exactly how computers operate, following instructions linearly. For example when a card is played this can easily be broken down into some function calls that can be made, firstly any effect the card might have on play must be enacted so a function called playedFunct() can be called to do this, then the card must be removed from the list of cards in your hand then added to the list of cards on the board, this can be done via a simple function call or even a single line statement which can pop and append the card (though one-liners like this can be disadvantageous when debugging as they can often seem more logically complex than they actually are). And finally the card has to be rendered on the board so a function renderCard(card,position) can be called, allowing the card to become visible on the board. Every facet of the gameplay can be broken down logically as such, as can the menus outside of the game.

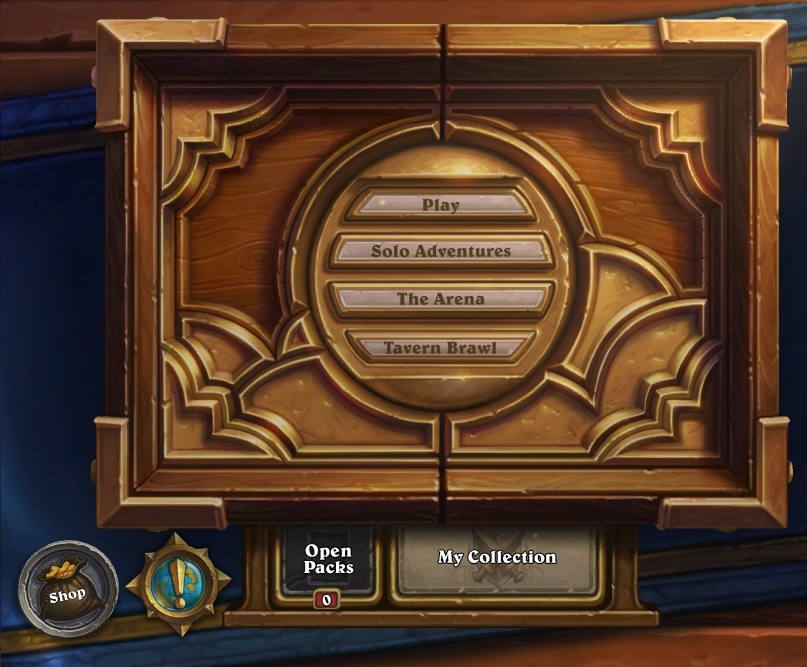
The reinforcement learning based AI which will be playing the game is not only best suited to computers as it can interact linearly with the game, it is impossible to make without them as you would need some other device capable of intelligent decision-making to be playing the game, which is limited to only humans and computers. Using a human here would defeat the purpose of having an AI opponent as the game is meant to be playable by yourself. The AI’s training makes use of iterative self-improvement (iteration being one of the most key computing concepts), performing mathematical and logical operations to achieve this at a speed not physically achievable by non-computational methods. The AI operates on a basic level by taking data from playing the game and optimizing itself so that it can create an output which will be the most likely to secure it the ability to win the game, this satisfies the computational approach as it is simply enacting a mathematical function on the input data to create an output

The game will be made in an object-oriented manner as I find this to be the best method to create the game whilst maintaining an intelligible, logical, code flow. Each card for each player can be instantiated as an object of the class “Card” which allows the cards to have their own unique properties (such as a name, cosmetic text, health, cost, attack, etc.) and were I not using an object oriented structure like this it would be much more difficult to logically allow the game to function. This process can be shown to be effective when using the example of playing a card again as using objects the cards themselves can have a .play() function, which is important as if it was necessary to use a global function the player number would need to be specified as each player has cards and the syntax within this function would be much more complicated, requiring many IF statements to determine which player is playing and what card they’re playing. Once the card played is determined when using objects any effects the cards may have on other cards can be easily enacted as they can directly adjust the properties of the other cards, but when not using an object oriented approach there are no individual card properties to edit, and as such everything would have to be kept in lists and adjusted this way, resulting in much more unreadable code, greater difficulty in adding more cards and inefficient code.

 *Below is a mock-up of the Card class structure needed to play a card:*

## Research

When looking into creating a game such as this I must first look at other similar games to see what inspiration I can derive from them but also more importantly what I can improve on comparatively to them. As part of my research I played and looked at the features of the two most popular online card games (Hearthstone and Magic: The Gathering, Arena) and the recently released DOTA Underlords, which is not a card game but uses an interesting unit buying system which I feel can help solve the problem of card drawing randomness which often negated some amount of the skill involved in gameplay for the first two games I looked into.

 Firstly I looked at Hearthstone, which is a game which is designed to look as smooth and user-friendly as possible, meaning it is an automatic attractor for those newer to the card game community or those who appreciate the polish put in to it.

*The Hearthstone main menu screen*

Hearthstone’s simple user interface is extremely appealing and as such I have decided to use a more simplified visual main menu structure rather than using text or any more complicated layouts which I feel needlessly confuse the user. The game has a pack opening system and an overarching card “collection” which indicates the cards which you do and do not own, and therefore the cards which you can and can not put into decks. I feel this is unfair to the end user and instead have opted to allow each user to have access to all the cards in the game.

The game’s concept is relatively simple, you draw 3 or 4 cards at the start of the game (dependant on whether you go first or second), you start at 1 mana crystal, gaining one per turn until you have 10. Mana crystals can be spent by playing cards and will refresh at the start of each player’s turn. There are two main card types, minions and spells. Minions are creatures which are summoned on to the board which can fight for you, either attacking your opponent or their minions. Spells are instant effects which do not necessarily summon creatures but can affect them or either player’s character. When the first player reaches 0 health the game ends.

I have decided to simplify cards into one type instead of using minions and spells. Cards will always summon a creature and can have effects similar to those of hearthstone’s spell cards attached to them, thus meaning the game is more easily understood. I have also opted to use a similar combat style, however unlike in hearthstone, you must always attack the creatures on the board before attacking the opposing player, allowing the game to last for longer and for creature on creature combat to be a larger facet of the game.



*A standard board state in Magic: The Gathering, Arena*

The second game I researched was Magic: The Gathering, Arena. Magic: The Gathering is one of the longest standing card games ever made and its advent into the market of digital card games has given me an opportunity to see what works and what doesn’t comparatively to Hearthstone.

The game functions slightly differently to hearthstone in that instead of mana crystals you gain as the turns advance you have the ability to play “land” cards which allow you to play other cards by spending them each turn. It is also different in the fact that your opponent gets to dictate the path on which your attacking creatures attack, meaning the game is also reliant on bluffs and not knowing your opponent’s decision-making.

I believe that whilst this does promote the skillful aspect of the game that my stakeholders want, bluffing and strategies such as this are almost impossible against an AI opponent as it does not make decisions in the exact same way humans do and as such I am allowing the player full control of their creatures’ attacks. In Magic: The Gathering damage inflicted to creatures is not permanent due to the game originating in physical form so it was nearly impossible to keep track of damage inflicted to so many things. Since my game will be digital the damage inflicted to creatures will last between rounds as computers can keep track of this very easily.

 One feature of Magic: The Gathering that I wish to implement if I have time is the ability to be able to stack like units on top of one another to remove clutter on the board and allow the game to look more visually appealing, which I believe is a key factor in player accruement.

*A standard board state in DOTA: Underlords, with purchased units displayed at the bottom*

The final game at which I looked for my research was DOTA: Underlords, a recently released game based on the popular mod DOTA: Autochess. This was the only game I researched that was not a card game, but rather just a strategy game in a more broad sense to see if I could incorporate any features currently unused by digital card games but applicable to strategy games in general to allow mine to stand out.

One particular feature that stood out to me while researching this game is the unit purchasing system in which each round you are offered a selection of units which you can buy with gold you collect over the course of the game, when applied to a card game instead of drawing a card each turn I feel that this would be extremely beneficial to the skill requirement of the game and thus the long term enjoyment levels that can be achieved. Adding this feature also removes the need for deckbuilding, which becomes simply a game of statistics instead of an interesting facet of the game with which to play due to the easily analysable gameplay of card games.

## Stakeholders

When looking into stakeholders for my game a number of my peers expressed interest in such a game given certain features they wanted were implemented. One particular interested party is a friend of mine named Vishaan who has a background in games that involve longer term strategy so as such a card game was a natural attractor for him. The game perfectly suits his desire for strategizing as there are nigh on infinite possible iterations of the game and its state so a good understanding of strategic thinking is rewarded.

He also enjoys games with a high degree of customization as evidenced by his investment in RPGs with character customization in the past, so the deck customization that is possible appeals to him as well. He hopes that in the way the game is coded it will be easy to add new cards to allow the game to feel fresher as new cards can rejuvenate a game to ensure it’s long term enjoyability. He also prefers games in which the opponents feel more real and as such I have decided to use reinforcement learning AI as opposed to the standard model of using strings of conditional statements to make the decisions the opponent makes less rigid.

Another potential interested party is my younger brother Micah as he has enjoyed digital card games in the past but feels like the current available market for them is too bare for his liking, as such he wants a game with different gameplay to the other two major online card games of Hearthstone and Magic: The Gathering, Arena. In order to achieve high enough difference from these games he feels that the game I am making should have mechanics with more choice to replace the classic method of drawing cards implemented by other games which can lead to a high amount of luck being involved in gameplay as opposed to pure skill so as such I have decided to instead opt for a turn based card buying system instead which, whilst it still has some variance, rewards skill over luck a greater proportion of the time. He has also requested that the game have a screen that will scale to any screen size automatically so it does not take any additional effort to use it on different devices and as such the game will be able to be enjoyed by a wider range of device owners who can now use any device without the game looking any different between them.

The third stakeholder I have identified is another friend of mine who very much enjoys gaming but is not currently very involved in card games or strategy games in general and as a result of this lack of current investment into games of this type he has said he would enjoy it if the game were to have an explanation of the rules included within it to make his learning process easier. In order to fulfill this request I have decided that I will add a basics tutorial section to the game accessible via the main game menu so that the game can be picked up by new players with greater ease.

### Interviews

I conducted short interviews with my stakeholders in order to identify what exactly they wanted to see from the finished product of the game:

My first interviewee was Vishaan, below is a record of the interview that was conducted:

Me: What, for you, makes a game enjoyable?

Vishaan: Well, in my opinion as long as the game is of a type that I like it just has to not be completely reliant on luck but have enough variation so it doesn’t feel stale

Me: How do you think one could tackle the problem of luck versus variance in a card game?

Vishaan: Im not 100% sure, but if I had to guess I would say I would want not to have to build a deck of cards so the games I play don’t always feel the same for each deck

Me: Alright, and what do you feel, if anything, that current card games are doing wrong that I could improve on?

Vishaan: Yeah there is one thing that annoys me about card games right now, most games do this actually and its really offputting. Its that their AIs for enemies don’t feel like they’re actually a challenge and I think if they seemed more human then I could look at them as more of a valid opponent

## Hardware Spec

The game will not be extremely hardware intensive as all the AI’s training will be done on my computer beforehand so the intense training process is not necessary to carry out on the end user’s machine. If someone were to attempt re-training it or training an entirely new neural network to use instead of mine (this can be done relatively easily and would normally be done to adjust the difficulty level of the AI opponent as less training would mean more difficulty and vice versa up to a point) I would recommend a system using:

* Dedicated GPU with no less than 4GB of VRAM
  + GB+ of VRAM would be recommended but can be very expensive
  + A more powerful GPU is more impactful on training performance than a more powerful CPU
* CPU with at least 4 cores with a clock speed of 3GHz
  + Improving the CPU will increase the performance of training with more cores being more impactful than higher clock speed but an upgrade of CPU will almost always be worse than an upgrade of GPU

These hardware specs are important in the event of a retraining in order to attempt to train it in a practical amount of time (i.e. less than a week). The only real requirement when you are not retraining is that the hardware is able to support all the necessary software at a basic level for the computer to run it which, according to Intel (<https://software.intel.com/en-us/distribution-for-python/system-requirements>), Is at least one gigabyte of secondary storage (SSD or HDD are both acceptable as the read/write speed will not affect the performance very much) for the base python and an additional 2 to 3 gigabytes for the rest of the libraries that need to be imported such as pytorch.

The system should also have:

* A processor at least as powerful as a single core 1.6GHz processor
  + This is normally the power level of the lowest price point ARM or Intel Atom processors
* Standard peripheral devices required for computers to function normally
  + At least one monitor on which to see the gameplay
  + A keyboard (even 80% size keyboards still work)
  + A mouse

## Software Spec

There are relatively few pieces of required software for the game:

* A python version of 3.0 or more recent
  + Syntax changes heavily in some abundant features like print or input from 2.7 to 3.0 and as such the code would not be able to run on any older versions of python
* The most recent versions of the python libraries I use
  + Pygame
  + Pytorch
  + Random
  + Pprint
  + Math
* An operating system which supports all the libraries I have listed (i.e. the standard operating systems of Windows, most Linux distributions or Mac OS)

## Requirements

The game, upon completion, should have:

* Necessary Features for the game to function:
  + Card buying system
  + At least 10 different card types
  + Cards which trigger effects on:
    - Death
    - Play
    - Attack
    - End of turn
  + Ability to play cards from your hand onto the board
  + A limit on the space on your board and in your hand
  + Tutorial section
  + Ability to attack using your played cards to damage the opposing player
  + AI implementation of the second player
  + Ability to win or lose the game given certain criteria have been fulfilled
* Important features the game should have but are not necessary for it to function:
  + A menu with which to navigate, at the very least containing the ability to:
    - Play a game
    - Adjust your settings
    - View the tutorial
  + Ability to adjust the position of cards on your board
  + Ability to combine multiple of the same card into a single more powerful version of themselves
  + Ability to pause the game
  + Menu of all the cards and their combinations
    - Given enough time this could also be an interactable visual menu, though this is more of a stretch feature
* Features that would be nice to have but less likely to be finished:
  + The ability to retrain the AI from within the game menu
  + Different game modes to the standard mode of play
  + Local multiplayer versus another human player
  + In-game custom card creation feature
  + Ability to save the game state and return to it later
  + Animations for things like playing cards
  + Passive effects for played cards
  + Stacking like units

## Limitations

Whilst the game will be feature-rich and strategically in-depth as I can make it, there will of course be limitations to the things that I will have been able to complete or add to the game within the time frame I have to make it.

Such limitations include:

* No sound
* No custom animations for different effects
* No online or LAN multiplayer
  + I might possibly implement local multiplayer, however online or LAN multiplayer is much too complicated to do within such a time frame
* Only English language support

# Design

# Implementation

# Testing

# Evaluation

Analysis

Design

Implementation

Testing

Evaluation