UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Informática

Diseño y desarrollo de un sistema de adquisición de datos de partidas de Magic: The Gathering

Trabajo Fin de Grado

Grado en Ciencia de Datos

AUTOR/A: Pérez Muñoz, Antonio

Tutor/a: Morillas Gómez, Samuel

CURSO ACADÉMICO: 2023/2024

Resum

'Magic The Gathering' és un joc amb un gran nivell d'abstracció i dificultat, al qual moltes persones els pot portar fins i tot mesos aconseguir comprendre amb un nivell mitjà d'enteniment. Per això ens agradaria en aquest Treball de Fi de Grau idear i dissenyar una forma de representar els diferents estats de joc que poden existir.

Com a mostres utilitzarem partides completes registrades a través de Magic The Gathering Online (mtgo). El fi últim seria construir un model que rebi com a input els diferents estats pels quals ha passat la partida, i com a output rebem la jugada següent a realitzar en base a l'aprenentatge inferit de les mostres. Tot i que els nostres esforços es concentraran a construir tota una infraestructura que ens permeti adquirir i representar l'estat de joc dins d'una partida, la qual cosa ja és un problema extremadament complex degut a la gran quantitat de variables a tenir en compte.

Per dur a terme aquesta tasca, utilitzarem eines de multitud d'àmbits, com ara el reconeixement d'imatges i formes, per trobar i detectar cadascuna de les cartes, lectors òptics per poder extreure certes informacions rellevants que apareixen a la pantalla, algunes eines online com APIs per extreure la informació complementària necessària i algun tipus de model de generació del llenguatge per dur a terme el darrer pas de crear el nostre recomanador de jugades.

Paraules clau: Magic The Gathering, Magic Online, Modelització de jocs, Representació de dades, Extracció de dades

Resumen

'Magic The Gathering' es un juego con un gran nivel de abstracción y dificultad, al que muchas personas puede llevarles incluso meses lograr comprender con un nivel medio de entendimiento. Por esta razón nos gustaría en este Trabajo de Fin de Grado idear y diseñar una forma de representar los diferentes estados de juego que pueden existir.

Como muestras utilizaremos partidas completas registradas a través de 'Magic The Gathering Online'(mtgo). El fin último sería construir un modelo que reciba como input los diferentes estados por los que ha pasado la partida, y como output recibamos la siguiente jugada a realizar en base al aprendizaje inferido de las muestras. Si bien, nuestros esfuerzos se concentrarán en construir toda una infraestructura que nos permita adquirir y representar el estado de juego dentro de una partida, lo cual es ya un problema extremadamente complejo debido a la gran cantidad de variables a tener en cuenta.

Para llevar a cabo esta tarea, utilizaremos herramientas de multitud de ámbitos, como el reconocimiento de imágenes y formas, para encontrar y detectar cada una de las cartas, lectores ópticos para poder extraer ciertas informaciones relevantes que aparecen en pantalla, algunas herramientas online como APIs para extraer la información complementaria necesaria y algún tipo de modelo de generación del lenguaje para llevar a cabo el último paso de crear nuestro recomendador de jugadas.

Palabras clave: Magic The Gathering, Magic Online, Modelización de juegos, Representación de datos, Extracción de datos

Abstract

Magic The Gathering is a game with a high level of abstraction and difficulty, which many people can take months to understand with an average level of understanding. For this reason we would like in this Final Degree Project to devise and design a way to represent the different game states that may exist.

As samples we will use complete games recorded through 'Magic The Gathering Online' (mtgo). The ultimate goal would be to build a model that receives as input the different states through which the game has passed, and as output we receive the next move to make based on the learning inferred from the samples. However, our efforts will focus on building an entire infrastructure that allows us to acquire and represent the game state within a game, which is already an extremely complex problem due to the large number of variables to be taken into account.

To carry out this task, we will use tools from a multitude of fields, such as image and shape recognition to find and detect each of the cards, optical readers to be able to extract certain relevant information that appears on the screen, some online tools such as APIs to extract the necessary complementary information and some kind of language generation model to carry out the last step of creating our move recommender.

Key words: Magic The Gathering, Magic Online, Game modeling, Data representation, Data extraction

Índice general

Índice de figuras

CAPÍTULO 1
Introducción

Desde el principio de los tiempos, el ser humano siempre ha tenido interés en poner a prueba su intelecto, de ahí el surgimiento de los primeros juegos de mesa desde hace más de 8000 años[1].

Los primeros juegos de mesa tenían dinámicas sencillas y basadas casi exclusivamente en el azar, no siendo hasta los egipcios que se introdujo el tablero y por lo tanto un espacio asociado al juego y unas fichas que transitan por él, creando el juego popular conocido a día de hoy como el 'Senet'[2]. Serían posteriormente los romanos quienes darían una vuelta de hoja al asunto y popularizarían el 'Ludus latrunculorum'[3], un juego basado en estrategía militar como los que se desarrollarían en la edad media, el ajedrez y el Talf.

Como podemos darnos cuenta el transcurso del tiempo, al igual que ha pasado con muchos otros campos y ámbitos, los juegos de mesa han ido evolucionando y complicándose cada vez más, dando lugar a la aparición de lo que serían los juegos de mesa actuales a principios del siglo XX, con la aparición del 'Monopoly'. Esto nos traslada al año 1993, donde Richard Garfield[4], un doctor en matemática combinatoria, inventó el primer JCG (juego de cartas coleccionables) al cual denominó 'Magic The Gathering'. Esta idea surgió tras una conversación sobre el poker, donde se planteó como sería un juego de estrategia en el que en lugar de haber 54 cartas predefinidas, fueras tú quien construyera tu propia baraja. Esta idea sería posteriormente comprada por la entonces recién fundada 'Wizards of the Coast', una empresa que acabaría lanzando el juego y convirtiéndose en una gigantesca compañía, conocida no solo por 'Magic' sino también por otras grandes marcas como las ediciones tercera, cuarta y quinta de 'Dungeons and Dragons'.

Con el paso de los años, Magic ha seguido evolucionando, cambiando y creando un circuito competitivo que dota a sus jugadores de un motivo por el que jugar, mejorar y progresar dentro del juego.

1.1 Motivación Personal

Desde una temprana edad entré en contacto con 'Magic', un juego con una dificultad muy alta en el que cada partida es diferente y cada pequeño detalle puede marcar la diferencia al final. Una mezcla entre estadística, resolución de problemas y gestión de unos recursos limitados, junto con una necesidad de leer el ambiente y las situaciones muy similar a la existente en el poker, dieron lugar a un juego que encontré tremendamente divertido e intenso a nivel tanto emocional como intelectual, una combinación perfecta.

Aún así, no fue hasta este pasado año que, después de ir y volver una forma más casual al juego, comencé a tomármelo de una forma más seria y competitiva. Desde en-

tonces me he llegado a clasificar a dos regionales europeos, torneos en los cuales el nivel ya es mucho más alto que dentro de competencias nacionales. Al mismo tiempo que me interesé de nuevo por el juego conocí mucha gente que comparte mi pasión, y gracias a ellos acabé redescrubriendo lo que hoy nos trae aquí. 'Magic Online' es una plataforma empleada por los jugadores con un corte más competitivo, permitiéndoles jugar 'Magic' de una manera remota desde un ordenador, y es en este contexto donde surge la idea inicial de este TFG.

Desde la aparición de los primeros ordenadores, el ser humano ha intentado que las máquinas aprendieran a jugar a los juegos humanos e incluso los superaran. Esta tendencia cogió fuerza desde los años 50 cuando Claude Shannon publicó su famoso artículo[5] acerca de un ordenador que podría jugar al ajedrez. A pesar de encontrarnos en una época totalmente diferente, donde la tecnología está avanzando a una velocidad vertiginosa y donde las IAs son capaces de jugar a todo tipo de juegos, el problema que vamos a enfrentar es similar a los problemas que Claude Shannon ya enfrentaba hace más de 70 años.

1.2 Objetivos

El objetivo que buscamos es sencillo en cuanto a concepto, encontrar una forma de modelizar un juego tan complejo como es 'Magic the Gathering', intentando obtener una representación lo más fidedigna posible de la infinidad de situaciones que podemos encontrar dentro de una partida.

Si bien el primer esfuerzo será un trabajo más teórico y conceptual, que solo se puede abordar si se tiene un alto nivel de conocimiento del juego, el marco en el que vamos a trabajar es 'Magic Online', poniéndonos como objetivo extraer la información que consideramos relevante de una partida de 'Magic' jugada en la plataforma digital.

Por lo tanto, nuestro fin no es otro que constuir un flujo que nos permita extraer una representación veraz de lo que sería una partida, y estructurar dicha información para que en un futuro se puedan entrenar todo tipo de modelos, utilizando esta herramienta para representar de la manera más eficiente posible la complejidad de las situaciones que encontramos en una partida de 'Magic'.

1.3 Estructura de la memoria

Como se puede observar a continuación, encontraremos una gran cantidad de capítulos. Dado que el tema escogido es bastante de nicho, nuestro estado del arte estará muy acotado por la falta de trabajos acerca del tema. Utilizaremos el segundo de los capítulos encontrados a continuación para poner en contexto toda la problemática que vamos a encontrar, facilitando así que los lectores sean capaces de entender hasta cierto punto por qué se ha abordado el problema de dicha forma.

Una vez el lector tenga el contexto, nos centraremos en analizar en profundidad el problema que debemos abordar y, posteriormente, dispondremos de un apartado donde se explicará la solución finalmente aplicada. A continuación, se destinará un apartado para explicar los diferentes mecanismos utilizados para aumentar la robustez de nuestra solución. Para finalizar, encontraremos los capítulos en los que hablaremos del conocimiento extraído, evaluaremos los resultados, extraeremos conclusiones, propondremos trabajos futuros y encontraremos las referencias.

CAPÍTULO 2
Estado del arte

El tema escogido y sobre el que trabajaremos es un tema muy de nicho, por lo que no existen muchas iniciativas similares o cercanas. En los útlimos años, probablemente el único artículo que podemos encontrar que trata de hacer algo similar a nosotros es el escrito en 2019 por Austin Herrick, Alex Churchill y Stella Biderman[6], el cual se demuestra la alta complejidad del problema que vamos a abordar. En su artículo, los autores muestran que jugar de manera correcta a 'Magic' es al menos tan difícil como el conocido 'Halting Problem'[7], haciendo una propuesta para solucionar el problema que ha estado abierto por décadas, haciendo uso de un máquina de Turing arbitraria.

Mientras que en otros se exploran enfoques mucho más rudimentarios y antiguos, como es el caso de utilizar un conjunto de reglas construidas de manera manual[8]. Por otra parte, el resto de referencias que podemos encontrar generalmente son artículos de una temática totalmente distinta al presente TFG, enfocando el juego desde una visión de marketing[9] y sociológica[10], o enfoques acerca de como construir barajas de 'Magic' [11] [12].

Debido a esta falta de trabajos anteriores que aborden el mismo problema que nosotros deseamos solucionar, este trabajo tendrá una gran componente de creatividad e innovación al vernos obligados a partir de cero en muchos aspectos.

Si bien no existe una gran bibliografía acerca de como abordar nuestro problema, podemos fijarnos en enfoques tomados para resolver problemas similares para otros juegos. Por ejemplo en el artículo de Huang[13] nos puede ayudar a entender cual es el estado del arte de modelos de machine learning capces de jugar a juegos de estrategia como podría ser el StarCraft II y como en solo 60 horas de entrenamiento son capaces de derrotar jugadores humanos.

Encontramos también algunas herramientas interesantes como la desarrollada por Daochen Zha y Kwei-Herng Lai [14], los cuales han desarrollado una plataforma en Python para la investigación y desarrollo de aprendizaje por refuerzo en juegos de cartas mucho más simples, como el Blackjack, Leduc Hold'em, Texas Hold'em, UNO, Dou Dizhu or el Mahjong.

Podemos encontrar también un artículo escrito por empleados de microsoft[15], en el que utilizan modelos de lenguaje largos para construir una plataforma que permita abordar problemas complejos de la teoría de juegos, permitiéndonos a nosotros explorar como abordan la toma de decisiones estratégicas utilizando IA.

Por último, nos gustaría mencionar el artículo escrito por J. Niklaus y T. Koller [16], los cuales se centran en el juego de la baraja suiza Jass, donde la coordinación dentro de los dos equipos opuestos es crucial para ganar. Siendo por lo tanto relevante un aspecto que en ningún momento hemos visto antes, que es el estilo de juego de los diferentes

jugadores, ya que la comunicación verbal está prohibida. Esto hace que el juego sea un candidato particularmente adecuado para continuar la investigación sobre IA en juegos de cooperación con información oculta. Analizando en profundidad la efectividad y las limitaciones de varios algoritmos de última generación (variantes de Monte Carlo Tree Search (MCTS) y Redes Neuronales Profundas (DNN)), lo cual nos puede resultar de interés para investigaciones futuras.

CAPÍTULO 3
Contextualización

Previamente se ha hablado de que Magic es un juego sumamente complejo[6], por lo que para poder seguir el desarrollo de este TFG resultará crucial entender algunos conceptos básicos dentro del juego, si bien no se profundizará en detalle en las mecánicas propias del mismo.

Haremos una breve conceptualización básica antes de entrar más en profundidad, 'magic' es un juego por turnos uno contra uno, cada jugador tiene una baraja propia, con la que jugará la partida, y cada una de esas cartas tendrá características diferentes. Por ejemplo el tipo más básico de carta son las tierras 3.4, son cartas de las cuales podemos poner en juego una desde nuestra mano en cada uno de los turnos, y nos permiten agregar el recuso más básico del juego, el maná, de una forma que explicaremos más adelante.

Existen cinco colores diferentes dentro de magic, por lo tanto el maná que podemos tener es de diferentes colores, por ejemplo, podemos tener dos manás, uno rojo y uno verde, o dos azules. Esto es relevante porque las cartas que no son tierras poseen un coste, en la figura 3.2 podemos ver en la esquina superior derecha que su coste es un maná verde y uno de cualquier color, por lo que para poder jugar esta carta tendremos que tener disponibles exactamente un maná verde y luego uno de cualquier color, rojo, azul, blanco, negro o incluso otro verde, si esto es así, podríamos en nuestro turno jugar esta carta pagando dicho coste.

Para realizar las explicaciones utilizaremos figuras extraídas de 'Magic Online', así como otro tipo de figuras de otras fuentes.

Para empezar, debemos tener en cuenta que estamos tratando con un juego que se juega uno contra uno y por turnos, cada uno de los jugadores tendrá las zonas y elementos que se comentarán a continuación.

Cada jugador tiene un número de cartas en la mano, las cuales son desconocidas para su oponente, y una biblioteca o mazo, que serán las cartas de su baraja que todavía no ha robado. Se dispone, a su vez, de un campo de juego donde irán a parar las cartas jugadas o que deban ser mantenidas en esta zona para posteriormente ser aprovechadas de alguna manera. Por otro lado, cada jugador tendrá su propio cementerio, una zona a la que irán las cartas que tienen un solo uso o que se econtraban en su campo de juego y han sido destruidas.

Por último, encontramos la zona de cartas exiliadas. Puesto que los jugadores pueden jugar algunas cartas que interactuan con sus cementerios, esta sería una zona restringida en la cual si una carta entra ya no existen formas de interactuar con la misma.

Además de todas estas zonas, debemos tener en mente que cada jugador comienza la partida con 20 vidas, un contador que si cae a cero implica que ese jugador ha perdido.

En la figura 3.1 podemos apreciar cada una de las zonas señaladas a modo de esquema resumen, aunque más adelante iremos señalando algunos elementos más en profundidad conforme los vayamos necesitando. Siendo cada número de la figura el siguiente elemento:

1. Cartas en nuestra mano.
2. Nuestro campo de juego.
3. Nuestro cementerio.
4. Nuestro total de vidas.
5. Número de cartas en nuestra baraja.

Figura 3.1: Captura de una martida jugada en 'Magic Online'

Ahora que se han expuesto las diferentes zonas que tiene el campo de juego, explicaremos las diferencias que podemos encontrar entre las cartas. Por un lado tenemos los **permanentes**, son cartas que como su nombre explica permanecen dentro de la zona de juego tras haberlos jugado, resultando de especial interés para entender cual es la situación de la partida. Dentro de los permanentes tenemos muchísimos subtipos, pero nos centraremos únicamente en dos, las **criaturas** y las **tierras**.

Las criaturas son cartas que como podemos apreciar en la figura 3.2, tienen un coste en la esquina superior derecha, el cual debemos pagar para poder ponerlas en juego, y una fuerza y una resistencia, las cuales son extremadamente relevantes ya que son una de las formas en las que podremos más adelante ir reduciendo el total de vidas de nuestros oponentes.

Figura 3.2: Ejemplo de carta de criatura, donde el 1 representa el coste que se debe pagar para ponerla en juego, el 2 la fuerza y resistencia de la misma (el primer número hace referencia a la fuerza y el segundo a la resistencia) y el 3 el tipo de carta.

Las tierras, por otra parte, no poseen un coste y cada jugador tiene la posibilidad de jugar una de ellas en cada uno de sus turnos.

A diferencia de estas cartas, podemos encontrar cartas que no son permanentes, como sería el caso de los **hechizos**, y dentro de los cuales se pueden diferenciar los **instantaneos** y los **conjuros** (como puede verse en la figura 3.3). Ambos tipos de carta se juegan por el coste que encontramos en la esquina superior derecha, pero a diferencia de los permanentes, una vez jugados van directamente al cementerio.

Para entender mejor las diferencias entre instantáneos y conjuros, tendremos que profundizar un poco en el funcionamiento del juego. Si bien estamos hablando de un juego por turnos en el que cada jugador tiene su turno y este no finaliza hasta que no decide darle turno al rival, existe dentro del juego la posibilidad de realizar acciones en cualquier momento, aunque sea el turno del rival. De ahí nace la diferencia entre los instantáneos y los conjuros, ya que los primeros se pueden jugar en cualquier momento (aunque no sea tu turno), mientras que los segundos, al igual que las tierras, criaturas y otro tipo de permanentes, solo pueden ser jugados cuando es tu turno, y solo en fases concretas del mismo.

A esta complejidad basal que muestra este TCG, hay que sumarle un gran número de mecánicas que no hemos explicado como pueden ser las diferentes fases del juego, el funcionamiento del combate (el ataque y defensa con criaturas)... La existencia de efectos que pueden ser jugados en cualquier momento da lugar a la posibilidad de que un jugador pueda responder a las acciones del otro *antes* de que estas ocurran, dando lugar a la **pila**, un concepto muy similar la pila utilizada en programación, y muchísimas otras situaciones. Al final, el abanico de posibilidades de un jugador es muy grande, conformando un juego muy rico, pero a su vez esta capacidad de juego complica enormemnte las tareas de representación del mismo, ya que esta representación debe recoger un gran número de interacciones y decisiones.

Antes de dar paso a la siguiente sección, sin embargo, falta explicar un concepto de manera superficial que resultará clave más adelante: todas las cartas en el campo de juego tienen pueden encontrarse de dos maneras independientemente de su tipo, 'giradas' o 'enderezadas'. Las cartas 'enderezadas', tras ejercer su función, quedan 'giradas' y ya no pueden actuar hasta el siguiente turno. Este concepto nos interesa especialmente, ya que

Figura 3.3: Comparación entre una carta de hechizo instantáneo (número 1) y una carta de hechizo conjuto (número 2).

las tierras enderezadas, al girarse, nos agregarán un recurso, llamado 'maná', que nos ayudará a pagar los costes de las cartas.

Una tierra que ya se encuentre 'girada' no nos puede agregar ningún recurso, mientras que una tierra 'enderezada' nos puede agregar el maná que la propia carta indique al girarse, como se puede ver en la figura 3.4, donde esta tierra en concreto nos agregaría un maná azul. Algo similar ocurre con las criaturas, las cuales mientras están 'enderezadas' pueden bloquear un ataque de nuestro rival en su turno, o atacarle si es nuestro turno. Por lo tanto, debemos tener siempre en mente que no solo es importante qué cartas están en el campo de juego, si no si están 'enderezadas' (pueden actuar) o 'giradas' (no pueden actuar).

Figura 3.4: Ejemplo de carta permanente, donde 1 indica que es de tipo Tierra y 2 indica que el maná que se agregará una vez girada será azul.

Puede ayudarnos más adelante entender de una manera simplificada y vaga cómo funciona el combate. A modo de simplificación, nos hará falta saber que si tenemos una critura 'enderazada' en nuestro turno, en un momento concreto del mismo podremos

realizar la acción de 'girarla' con la intención de inflingirle a nuestro rival. En este caso, nuestro rival perdería tantas vidas como fuerza tiene nuestra criatura. A su vez, las criaturas 'enderezadas' de nuestro rival podrán ponerse en frente de una de nuestras criaturas para bloquear ese daño. Aunque esta es una simplificación extrema de cómo funciona la fase de combate dentro de un turno, la cual puede llegar a extenderse hasta 5 fases, será suficiente para entender algunas cosas que se mencionarán posteriormente.

Antes de pasar al siguiente punto, hemos considerado de interés hablar de un concepto un poco abstracto como es el **hacer objetivo**. Para entenderlo, lo más facil será hacer uso de un ejemplo, como es el caso de la figura 3.5, podemos ver señalada la parte del texto que dice 'al jugador objetivo', terminología que podemos encontrar en muchos efectos a la hora de jugar. Esto hace referencia al objetivo sobre el cual se lanza ese hechizo, en este caso, el efecto de ese hechizo lo sufrirá el jugador que quien juegue la carta designe como objetivo.

Figura 3.5: Ejemplo de carta donde se hace objetivo del efecto de la misma a un jugador únicamente

De todas formas, no debemos sentirnos abrumados por la información anterior, ya que los conceptos explicados son más que suficientes para comprender el enfoque que queremos hacer del problema.

CAPÍTULO 4
Análisis del problema

Tras la contextualización realizada en el capítulo anterior, ahora somos más capaces de hacernos una idea de la magnitud del problema al que nos enfrentamos. A esto se le suman diversos factores que nos limitan y pueden dar problemas a la hora de proponer diferentes soluciones.

- El primer paso que debemos resolver es ser capaces de extraer las cartas que tenemos en la mano.
- En segundo lugar, queremos saber cuales son las tierras que tenemos en juego, y si están 'enderezadas' o 'giradas', ya que nuestra capacidad para agregar maná será un recurso limitante a la hora de poder o no jugar una carta en concreto de nuestra mano.
- En tercer lugar, deberíamos encontrar una forma de modelizar la zona donde se encuentran las criaturas, también conocida como campo de batalla, tanto las propias como las de nuestro oponente.
- Identificar cual es el total de vidas de los jugadores resultará imprescindible a la hora de tratar de modelizar el estado de la partida.
- Por último el número de cartas restantes en nuestro mazo, mano, cementerio y exilio puede resultar muy interesante, mientras que la información concreta de qué cartas se encuentran en los cementerios y en el exilio tendrá una relevancia mucho menor.

Esto sería el conjunto de información básica que deberíamos tener en cuenta para ser capaces de modelizar un estado de la mesa. Dicho de otro modo, si vieramos un momento concreto de la partida, estos serían los factores más determinantes a la hora de tomar la siguiente decisión.

Con este primer vistazo general a nuestro problema, queda patente que queda por incluir mucha información compleja de modelizar. Ver todos los factores comentados anteriormente en un momento concreto de la partida nos puede ayudar a tener cierta idea de qué es lo que ha ocurrido previamente, pero tendríamos que intuir muchos otros factores. En esta situación, sería ideal tener un registro de cómo va evolucionando el estado de la mesa a lo largo de la partida, lo que no solo nos permitiría tomar las decisiones en consecuencia a lo que estamos observando, si no también a lo que hemos ido observando a lo largo de la partida.

A esto debemos sumarle que hay mucha información que no estamos registrando ni plasmando de ninguna manera: qué objetivos hacen las cartas que se hayan podido

jugar, cómo han sido los combates llevados a cabo a lo largo de la partida, o qué cartas hemos decidido jugar, puesto que si se tratara de instantáneos o conjuros no se reflejarían en ninguna de las partes que hemos mencionado anteriormente que tenemos interés a excepción del cementerio.

Como podemos apreciar, nuestro principal problema no solo es tener que ser capaces de representar todas las cartas que cualquiera de los dos jugadores vayan a jugar, si no que debemos tener en cuenta también todas las posibles interacciones que un jugador pueda tener con el otro (por ejemplo, jugar un instantáneo en respuesta a otra carta jugada por el otro jugador, hacer objetivo con una carta a otra de las cartas en la mesa...).

La cantidad de información, decisiones, objetivos y maneras de interactuar de los jugadores entre ellos, hacen la tarea de representar este juego de una forma fideligna algo muy complejo, razón por la cual se han tomado las decisiones que se explicarán en el siguiente capítulo.

4.1 Solución propuesta

Tras mucho estudio y reflexión, hemos concluido que tratar de modelizar cada uno de los posibles eventos, interacciones y decisiones que pueden ocurrir o han ocurrido dentro de una partida es una tarea con una embergadura inviable para este TFG. Sin embargo, hemos tratado de encontrar una solución más sencilla, algo más simplificada pero que nos dote de la suficiente información como para permitir en un futuro construir un modelo que sea capaz de aprender de las partidas previamentes grabadas haciendo uso de nuestra solución.

Aunque en este trabajo no trataremos de construir y entrenar una arquitectura para aprender a jugar a partir de una gran cantidad de partidas,dada la limitación de tiempo y de recursos exitente, sí que nos gustaría hacer una recomendación antes de empezar a explicar el enfoque que seguimos en nuestra solución. Dada la naturaleza de la solución, recomendaríamos el uso futuro de algún modelo de generación de lenguaje al que se le pueda añadir como input adicional vectores de longitud fija con información relevante, de esta manera el modelo resultante será capaz de procesar los diferentes estados de juego existentes a lo largo de la partida y dar una respuesta de cual sería la jugada que recomendaría realizar a continuación.Una vez dicho esto, entremos en materia.

El primer problema a abordar será como codificar las cartas que se encuentran en la mano, para ello hemos decidido que la mejor opción es partir de una información de la que ya disponemos, es decir, qué cartas tenemos en nuestra baraja y cuántas copias de cada una, de esta manera podremos codificar las cartas de nuestra mano como un vector de longitud n, donde n será el número de cartas distintas existentes en nuestro mazo.

En la figura 4.1 podemos ver un ejemplo de lista de cartas en el mazo a la izquierda, en texto plano, mientras que a la derecha podemos ver un ejemplo de como se codificaría una mano, en este caso en nuestra mano tendríamos una copia de la primera carta de la lista, una de la cuarta carta y 3 copias de la quinta.

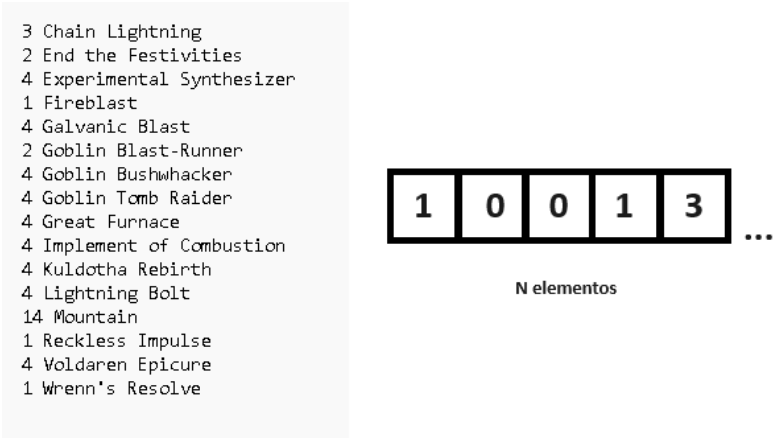

Figura 4.1: Ejemplo de una mano, donde a la izquierda se muestran las cartas en el mazo y a la derecha la mano codificada, siendo N el número de cartas distintas en nuestro mazo.

Una vez resuelta esta parte del problema, podemos pasar a cómo codificar las tierras que tenemos en el campo de juego. Puesto que generalmente lo más importante es saber qué tierras tenemos nosotros en el campo de juego, y si estas están 'enderezadas' o 'giradas' hemos decidido finalmente no incluir cuales son las tierras que nuestro oponente posee en juego. Con esto pretendemos construir una solución lo más robusta y fiable posible, como veremos en el siguiente capítulo en el que exploraremos en detalle cómo se han logrado los objetivos a nivel técnico, y la metodología elegida y utilizada no nos permite identificar las tierras de nuestros oponentes con el nivel de fiabilidad que deseamos.

Teniendo todo esto en cuenta, en un principio nos planteamos dos tipos de soluciones. Por un lado podemos identificar qué tierras tenemos en juego 'enderezadas', ya que estas son las únicas que pueden agregar maná, y por lo tanto son aquellos recursos con los que trabajamos. Por otro lado, podemos ir un paso más allá y almacenar directamente qué tipos de maná nos puede agregar cada una de las tierras que tenemos 'enderezadas'. Hemos optado por la segunda opción, por lo que codificaremos nuestras tierras como un vector de los seis tipos de maná que nos pueden agregar las tierras y la cantidad de cada uno. A continuación, podemos ver una representación visual de lo explicado anteriormente 4.2, en la que en una situación dada nuestras tierras podrían agregar dos manás rojos y uno azul.

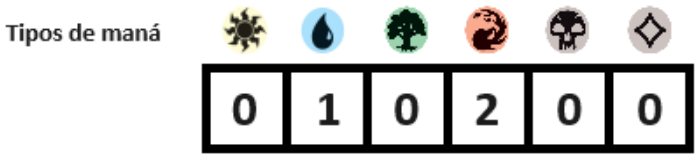

Figura 4.2: Ejemplo de tierras codificadas, donde nuestras nos agregarían un maná azul y dos manás rojos.

El total de vidas de cada jugador resultará una parte sencilla del problema, ya que simplemente nos tendremos que limitar a extraer cuál es el total de vidas en cada momento, dando lugar a una codificación en forma de un vector de dos elementos: nuestro total de vidas, y el de nuestro oponente.

El siguiente paso será cómo enfocar la codificación del campo de batalla en sí, y aquí nos encontramos con un gran problema, ya que en este caso tanto nuestra parte del campo de batalla como la de nuestro oponente son igualemente importantes. A esto debemos

sumarle que normalmente las criaturas suelen tener diversos efectos que pueden variar la fuerza y la resistencia de las mismas, por esta razón seguramente no sea fiable tratar de almacenar simplemente que criaturas se encuentran en el campo de batalla, si no que sería preferible almacenar exactamente cuál es la fuerza y la resistencia de las mismas en cada momento, por si hay algún tipo de efecto adicional que no estemos teniendo en cuenta.

Siguiendo este modelo de pensamiento, tenemos claro que puede ser mucho más interesante almacenar las estadísticas que las criaturas en juego tienen, en lugar de simplemente reconocer cual es cada una de esas criaturas. Nos enfrentamos a un problema común, ya que si queremos extraer una información tabular estructurada no podemos tener un número de variables indeterminado, y esta es precisamente la situación en la que nos encontramos, ya que el número de criaturas en juego va a ir variando a lo largo de la partida.

La decisión tomada para poder abordar de alguna manera este problema, aunque sea de una forma más simplificada y perdiendo algo de información, será constuir un vector de longitud fija cuatro. La razón es simple, si queremos abordar el problema que tenemos delante y mantener en la medida de lo posible toda la información que podamos, una solución sencilla y que retiene una gran parte de la información esencial será unicamente almacenar la fuerza total de todas las criaturas que controlan cada uno de los jugadores y realizar lo mismo para las resistencias. Esta no es la solución completa, pero es una forma que nos permite abordar este problema y retener una gran parte de la información.

Para comprenderlo mejor, podemos fijarnos en la figura, donde se aprecia el campo de batalla del oponente y el nuestro, y como aunamos la fuerza y la resistencia de todas las criaturas en cada campo de batalla siempre que estas estén enderezadas. En la figura 4.3 se puede apreciar como sería el funcionamiento, ya que de cada uno de los campos de batalla extraeríamos la suma total de fuerzas y resistencias de las criaturas 'enderezadas', lo cual daría finalmente lugar a un vector de longitud 4.

Figura 4.3: Ejemplo de campo de batalla codificado, con las dos cartas del oponente 'enderezadas' y nuestras cartas 'enderezada' y 'girada'.

En pos de simplificar el problema y que sea mucho más abordable, algunos datos de interés como son las cartas que un jugador puede tener en su cementerio o en su exilio quedan fuera del alcance de este proyecto, resultando suficiente con saber cuál es el número de cartas que cada jugador posee en estas zonas, de manera que la codificación será idéntica a la explicada anteriormente para el total de vidas de cada jugador. De esta misma forma, el total de cartas restantes en cada uno de los mazos y la cantidad de cartas que cada jugador tiene en su mano se enfocará exactamente igual. Por lo que todos los campos que acabamos de mencionar ocuparán un vector de longitud total ocho, cuatro para dichas variables bajo nuestro control y cuatro más para las mismas variables pero por parte del oponente.

Por último, nos enfrentamos a una tarea complicada, y es que debemos modelizar todas aquellas acciones que aún no se han tenido en cuenta, como son los objetivos de los hechizos, las jugadas realizadas por cada jugador (aunque en parte podemos descifrar esto de toda la información recabada), los ataques que cada jugador realiza...

Para enfocar esta tarea que parece inabordable, nos ayudaremos de una información valiosísima que el propio 'Magic Online' nos proporciona: el log. Este log nos permite visualizar cuáles son cada una de las acciones que se han llevado acabo a lo largo de una partida, por lo que engloba todos estos factores que no habíamos abordado antes. La única pega que se le puede atribuir es que solo posee la información de los hechizos jugados, los ataques realizados, y un registro de como se ha ido pasando de el turno de un jugador a otro.

Si se ha seguido todo el proceso, puede verse que la forma de enfocar este problema no es casual. En un primer lugar hemos explicado cómo abordar qué cartas tenemos en la mano, el maná que tenemos disponible y las vidas de las que dispone cada jugador, y podemos considerar toda esta información como los recursos de los cuales disponemos en el momento concreto de la partida. Por otro lado, información como el número de cartas en las manos, cementerios, exilio o las estadística de combate que posean cada jugador podría ser considerados como marcadores de cuál es la situación de la partida en un momento concreto. Por último, el log nos aporta la información acerca de que decisiones se han tomado a lo largo de la partida.

Si combinamos estas tres partes (recursos disponibles, situación de la partida y decisión tomada), tenemos una composición muy buena que nos permite modelizar con un alto grado de precisión cuál es la situación de una partida. Además, disponemos de una herramienta que puede extraer todas las variables que un modelo posteriormente pueda necesitar para realizar predicciones acerca de cual debería ser la siguiente jugada a realizar.

4.2 Plan de trabajo

Hemos encontrado de especial interés comentar brevemente como será la planificación de trabajo seguida para desarrollar este TFG.

El primer paso a realizar es definir correctamente cuales queremos que sean los **objetivos** que vamos a abordar con este trabajo, y una vez tengamos claros estos objetivos **trazar el plan** estimado para poder llevarlos a cabo. Estas tareas se estiman que tendrán una duración aproximada de 10 horas totales, que si bien puede parecer una gran cantidad de tiempo, resulta esencial ya que todo el tiempo de más que dediquemos a esta parte será tiempo que nos ahorraremos más adelante en el resto de las tareas.

El segundo paso de nuestra planificación debe ser **plantearnos las diferentes soluciones que podemos encontra**r para completar nuestros objetivos. En este punto debemos

realizar una investigación exhaustiva para estudiar qué tecnologías podemos utilizar para alcanzar nuestro objetivo, y evaluar los pros y contras de cada uno de los diferentes enfoques y herramientas. Por ejemplo, si una herramienta nos llevará demasiado tiempo aprender a utilizarla al nivel que necesitamos, seguramente sea mejor hacer uso de algo que nos sea más familiar y sencillo de usar en pos de ahorrar tiempo en esta ocasión. Para esta parte estimamos que se necesitarán 40 horas de trabajo. Es importante recalcar que una vez tengamos clara la solución que vamos a tratar de desarrollar seguramente tengamos que volver a este punto en algún momento, ya que conforme se desarrolle el proyecto deberemos explorar nuevas opciones que quizás en un principio no nos habíamos planteado, pero que resultarán ser más viables.

El tercer paso será **desarrollar la solución** que hemos decidido llevar a cabo. En este apartado es donde dedicaremos la mayor parte de horas, ya que tendremos que desarrollar nuestra herramienta para extraer la información y deberemos programar y hacer uso de todas las herramientas que previamente hemos considerado útiles para la consecución de nuestros objetivos. En nuestro caso concreto, este bloque de trabajo se ha desglosado en varias partes.

La primera parte será centrarse principalmente en trabajar solo con imágenes, desarrollando una batería de funciones para ir extrayendo toda la información que hemos detectado como útil. Se subdivide por lo tanto esta primera parte en: extraer las cartas de la mano, el maná disponible, las estadísticas de las criaturas, la extracción de la información que aparece en el log... El resto de variables descritas en capítulos anteriores se agrupan como una sola tarea, por lo tanto este primer bloque de trabajo con imagenes del 'Magic Online' se considera que puede llevarse a cabo en un total de unas 160 horas aproximadamente.

Una vez hemos sido capaces de extraer de imagenes estáticas la información de interés, podremos comenzar con el segundo bloque de trabajo: trabajar con los videos. Hasta este momento éramos capaces de extraer lo que sería una fila de información de cada imagen, pero ahora tendremos la tarea de detectar cuando hay alguna jugada en el juego. Debemos por lo tanto capturar de nuevo la información, construyendo así una nueva entrada en nuestra tabla resumen que almacenará todas las jugadas, recursos y estados de mesa por los que ha pasado la partida en órden cronológico.

Se realiza una estimación de que esta tarea puede llevar unas 40 horas de trabajo. Si bien ya tendremos todos los métodos para extraer la información de una imagen, deberemos ser capaces de detectar cuando se está realizando una jugada, así como construir un codigo 'main' en el que aunaremos todo el trabajo hecho previamente de manera independiente. Sin embargo, este proceso suele dar algunos problemas de compatibilidad y nos llevará a perder algo de tiempo reajustando algunos parámetros.

Una vez hemos terminado el código al completo y todo está funcionando correctamente, nos gustaría dejar unas 50 horas extras para monitorizar el funcionamiento del flujo y detectar los posibles errores que puedan aparecer, así como solucionarlos y realizar las modificaciones convenientes para obtener un código más robusto y funcional, más que una herramienta de juguete que solo funciones en unos cuantos casos de ejemplo.

Si bien el tiempo estimado para esta tarea es tan solo de 50 horas, se espera que sea algo mayor, ya que cuando se trata de trabajo de mejora y corrección de errores, lo normal es encontrarnos en una situación en la que podríamos dedicarle cientos de horas y seguir pudiendo realizar mejoras. En este último punto de trabajo incluíriamos también la utilización del código final para la extracción de las partidas que queramos utilizar como pruebas para estudiar el correcto funcionamiento del mismo.

Al mismo tiempo que todos estos procesos, hemos considerado de interés reservar unas 70 horas para dedicarlas a la escritura de la memoria, pudiendo redactar los capítulos pertinentes confome vayamos avanzando por las diferentes fases del proceso, y reservar una parte del tiempo para al final poder realizar todas las correcciones y modificaciones que sean necesarias.

4.3 Análisis del marco legal y ético

El desarrollo de un proyecto como el propuesto, que involucra la captura y representación de datos de partidas de 'Magic Online', plantea consideraciones importantes desde el punto de vista legal y ético. Estas consideraciones se centran en la recopilación, uso y procesamiento de datos, así como en las implicaciones éticas de la uitilización de dicha información.

En cuanto a los derechos de Propiedad Intelectual, debemos tener en cuenta que 'Magic: The Gathering' es una marca registrada de 'Wizards of the Coast'. Por este motivo, si queremos hacer uso de de imágenes o datos relacionados con el juego, debemos siempre hacerlo respetando los derechos de propiedad intelectual. Si en un futuro se quisiera hacer público este trabajo y el código asociado con el mismo, tendríamos que disponer de los permisos necesarios para utilizar el contenido del juego de manera apropiada.

En cuanto a la privacidad y el consentimiento de los usuarios, debemos entender que la herramienta desarrollada y que se explicará a continuación hace uso de partidas grabadas, por lo que si bien no se almacenan datos como podrían ser los nombres de los usuarios, si se deseara hacer uso del mismo, se debe tener en cuenta que en las grabaciones de las que se extraerá la información se podrá identificar a los usuarios por su nick dentro del juego.

En cuanto a las Normativas de Recopilación de Datos, si bien la adquisición y procesamiento de datos deben cumplir con las leyes de protección de datos vigentes, como el Reglamento General de Protección de Datos (GDPR) de la Unión Europea, nuestro flujo no almacena ningún tipo de información sensible ni que necesite de una anonimización, por lo que cumpliríamos con esta normativa.

A nivel ético, a lo largo de esta memoria intentaremos mantener siempre la máxima transparencia en cuanto al proceso de recolección de datos, tratando dejar siempre clara cual es la capacidad y el alcance del código desarrollado y que no exista ningún tipo de manipulación ni sesgo. El posible Impacto Social que puede tener este trabajo es mínimo, no considereamos realista pensar que este trabajo pueda impactar de una forma nociva a la comunidad de jugadores o a la industria del entetenimiento. De forma contraria, si algún jugador tuviera en un futuro interés por nuestra tecnología, esperamos que le pueda servir como un trabajo académico base sobre el que construir sus modelos o almacenar información de sus partidas, con la finalidad de convertirse en un mejor jugador.

CAPÍTULO 5
Solución aplicada

Con anterioridad se ha hablado sobre los problemas a los que íbamos a enfrentarnos al tratar de buscar una solución que se aproximara lo máximo posible a la realidad que buscamos representar, hemos ido dando ya poco a poco algunas pinceladas acerca de la forma en la que queremos que se encuentre la solución propuesta. Es ahora el momento de comenzar a profundizar un poco más y ver exactamente como hemos obtenido dicha solución y cuales han sido las herramientas de la cuales hemos hecho uso.

Para explicar todo el proceso que seguimos a la hora de extraer la solución esperada, hemos considerado como mejor opción seguir el orden en el que se desarrollo el proyecto, explicando en primer lugar como extraemos la información de una imagen extraída de una partida completa de 'Magic Online'.

Antes de comenzar, debemos dejar claro que trabajaremos utilizando una estrucura dataframe que posteriormente será exportada en formato csv. Esta estructura tabular tendrá las siguientes columnas: una columna por cada carta distinta que tengamos en nuestro mazo (que utilizaremos para codificar la mano), un total de cinco columnas para codificar la cantidad de mana de cada color que podemos agregar, ocho columnas para codificar el total de vidas, el número de cartas en los mazos, exilio, cementerios y mano de cada jugador, cuatro columnas para codificar la fuerza y resistencias totales que cada jugador tiene en el campo de batalla y por último una columna donde almacenaremos el log en formato de texto plano. Cada fila de este dataframe será una actualización de todos los valores almacenados tras la realización de cualquier acción por parte de cualquier jugador. Utilizando para el procesado de imagenes y videos varias librerías open source como son pillow[1] y openCV[2].

En la figura 5.1 podemos visualizar de manera esquemática la estructura de datos que vamos a seguir, teniendo tres bloques principales, modelización de los recursos de los que disponemos, situación en la que se encuentra la partida y la decisión que se ha tomado.

[1] https://pypi.org/project/pillow/
[2] https://pypi.org/project/opencv-python/

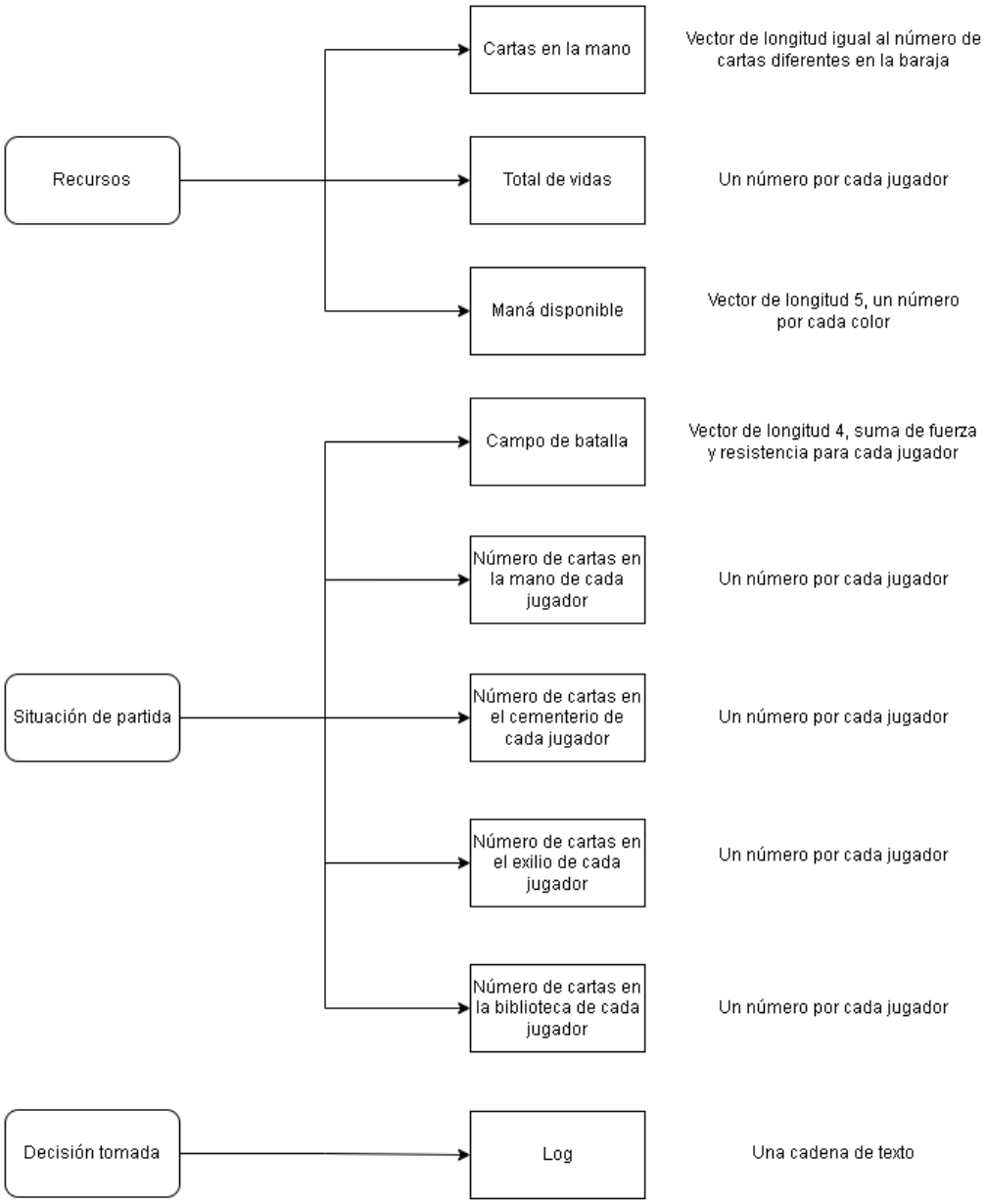

Figura 5.1: Estructura de datos.

5.1 Recursos disponibles

Comenzaremos por la modelización de los recursos disponibles, en concreto la codificación de las cartas en la mano. Como información previa, necesitaremos tener un listado con las cartas que tenemos en el mazo, y la estructura será sencilla, un fichero de texto plano en el que en cada linea del fichero aparecera un número, en referencia al número de copias de la carta concreta en el mazo, seguido de un espacio y el nombre de la carta.

Para realizar la extracción de información haremos uso de múltiples herramientas. La primera tarea a lograr es detectar de todo el espacio que es nuestra mano, qué partes son cartas y cuales simplemente fondo.

En primer lugar recortaremos de toda la imagen el fragmento en el que se encuentran las cartas de la mano, como podemos ver en la figura 5.2. En esta tarea concreta el borde negro que rodea una carta y el fondo son de una tonalidad muy similar, por lo que el mejor recurso que tenemos es extraer cada una de las cartas de manera manual. Aunque en la figura no se pueda apreciar bien, el tamaño de todas las cartas es el mismo, por lo que sabiendo cómo de grande es la ventana en la que se encuentran las cartas de nuestra mano y las dimensiones de cada carta, podremos segmentar la mano en una imagen por cada carta que tenemos en ella.

Una vez hemos separado la imagen de cada una de las cartas, hemos optado por seguir un enfoque de reconocimiento de texto. Durante un tiempo estuvimos planteando construir un modelo que pudiera reconocer cada una de las cartas en función de su aspecto visual. Sin embargo, esta idea tenía un problema, ya que una misma carta puede tener varios artes (o ilustraciones) distintos. Por lo tanto, para hacer la herramienta lo más robusta posible, la mejor opción sería recortar la parte superior de cada una de las cartas, y aplicar un OCR para reconocer los nombres de cada una de ellas. En concreto utilizaremos el lector óptico pytesseract[3][17] por ser de código abierto y ser una de las opciones gratuitas más competitivas del mercado. Una vez obtenidos los nombres de todas las cartas en nuestra mano, podemos rellenar facilmente las columnas del dataframe referentes a que cartas del mazo tenemos en la mano. Nos podemos dar cuenta de que esta metodología basada en un OCR puede tener cierto margen de error, por eso se ha implementado también métodos para asegurarnos de la robustez de este enfoque, que se podrán estudiar con profundidad en el capítulo siguiente.

Figura 5.2: Ejemplo de cartas que conforman una mano.

Antes de explicar cómo hemos extraído la información del siguiente recurso a modelizar, las tierras que tenemos disponibles, es importante recalcar que antes de poder llevar a cabo este proceso necesitaremos una lista con las tierras que jugamos en nuestro mazo, siguiendo la misma estructura que utilizamos en la lista de cartas del mazo.

Con esta información ya disponible, podemos comenzar con el proceso. Para explicarlo, será lo más sencillo hacer uso de un ejemplo como el que encontramos en la figura 5.3. La primera tarea será localizar las cartas que se encuentran en la zona de las tierras, ya sean 'enderezadas' o 'giradas'. Por lo tanto, cogeremos el segmento de la imagen en el que aparecen las tierras, una vez extraído el segmento de la imagen, pasaremos la imagen a una escala de grises y trataremos de enocntrar los contornos de cada carta dentro de la imagen umbralizada haciendo uso de un umbral(en el capítulo siguiente podremos entender como se modifica el valor del umbral según ciertas condiciones para poder extraer siempre las tierras de manera correcta).

Una vez tenemos detectados los contornos de cada una de las cartas nos topamos con un problema: si todas las tierras se colocaran unas separadas de otras, poríamos haber conseguido ya nuestro objetivo de extraer los contornos, pero si nos fijamos en la parte de la izquierda del ejemplo, podemos ver como al jugar dos tierras iguales una se superpone

[3]https://pypi.org/project/pytesseract/

a la otra. Para solucionar este problema, jugaremos con el tamaño que debe tener cada una de las tierras, los casos que podemos encontrar son los siguientes:

1.El contorno encontrado tiene un tamaño muy **similar** al tamaño de una carta, por lo tanto la imagen contenida dentro del contorno puede ser guardada para ser procesada, puesto que es solo una tierra.

2.El contorno encontrado tiene la altura de una tierra pero un **ancho notablemente superior** a la misma, por lo que nos encontramos en una situación en la que tenemos más de una tierra del mismo tipo una encima de otra, lo que haremos en este caso será seleccionar para almacenar solo la imagen de la tierra que está superpuesta a las demás, tarea sencilla, puesto que sabemos que dimensiones tiene. Contaremos cuántas tierras iguales tenemos colocadas debajo de esta, puesto que sabemos el tamaño de la tierra que se ve completa. También sabemos que todo lo que nos sobre de anchura hacia la derecha serán copias extra de la tierra, solo hemos tenido que estimar cuánto mide de ancho el trozo que se puede ver justo debajo y así podemos saber cuántas copias adicionales no hemos sido capaces de reconocer, es el caso que se puede apreciar en el ejemplo 1 de la figura 5.3.

3.Capturamos un contorno que **no se asemeja** a las dimensiones esperadas. Lo normal en este caso es que hayamos detectado una tierra que se encuentra 'girada', por lo que **no nos interesa** almacenar esta información, en el ejemplo 2 de la figura 5.3 se puede ver como una de las tierras está 'enderezada' y otra 'girada', por lo tanto la 'enderezada' sería detectada sin problemas, mientras que la 'girada' se desestimaría.

Figura 5.3: Ejemplos de la disposición superpuesta de dos tierras iguales al ser jugardas en la mesa. En 1 las dos cartas se superponen estando 'enderezadas', mientras que en 2 una de las tierras se encuentra 'enderezada' y la otra 'girada'.

Una vez encontrados todos los contornos que influyen, aplicaremos el OCR a la parte superior de las tierras que nos interesan y detectaremos de que tierras se trata, partiendo del hecho de que las cartas que están por debajo de otras serán siempre una copia extra de la tierra que tengan encima. Ahora que tenemos una lista con los nombres de todas las tierras que tenemos 'enderezadas' (teniendo en cuenta también las posibles copias extras que estén superpuestas por otra copia de la misma tierra), haremos uso de la API oficial de 'Magic The Gathering'[4], que nos permitirá acceder al texto que tiene una carta. De esta manera extraeremos el texto de todas las tierras de nuestro mazo, y haciendo uso de *regular expressions* filtraremos las expresiones que contengan 'Add' seguido de alguna letra, lo que nos indica que tipo de maná agrega esa tierra.

Como podemos ver, ahora que tenemos todas las piezas el proceso resulta evidente. Cogemos todos los nombres de las tierras 'enderezadas' y vamos anotando qué tipo de maná agregan dentro de la fila del dataframe que estemos escribiendo y la columna donde almacenemos ese color de maná correspondiente. De esta forma, cuando terminemos de recorrer la lista de todas las tierras 'enderezadas', ya sabremos cual es la cantidad y combinación de colores de maná que podemos agregar.

[4]https://docs.magicthegathering.io/

El siguiente paso será **extraer las vidas** que ambos jugadores tienen. Esta tarea como podremos observar es mucho más trivial que las mencionadas anteriormente. Dentro de una imagen simplemente recortaremos las zonas donde aparecen las vidas y, se aplicará un umbral de una forma similar a la realizada anteriormente. De esta manera, al aplicar el OCR reconoceremos cual es el número de vidas de cada uno de los jugadores y lo podremos añadir a las dos columnas correspondientes del dataframe. La localización de las vidas se puede ver en la figura 5.4,así como apreciar que el número a extraer está escrito en blanco, por lo que aplicar un umbral fijo funcionará especialmente bien a la hora de eliminar el posible ruido del fondo de la imagen que no nos interesa.

Figura 5.4: Ejemplo de mesa de juego en 'Magic Online'. En la imagen se pueden apreciar dos recuadros rojos a la izquierda, que resaltan las vidas de cada jugador.

5.2 Estado de la partida

Ahora que hemos terminado de analizar cómo extraemos los diferentes recursos a partir de la imagen de una partida, nos centraremos en explicar cómo se extrae el **estado de la partida**, comenzando por las criaturas en el campo de batalla. Para ello seleccionaremos dos segmentos del mismo, aquel en el que se encontrarán nuestras criaturas y aquel en el que se encuentren las de nuestro oponente. Todos los métodos que comentaremos a continuación se aplicarán sobre cada una de las partes de la imagen, lo que nos permitirá extraer la fuerza y resistencia de las criaturas de ambos jugadores.

Mediante un proceso de umbralización igual al que llevamos a cabo en el reconocimiento de los contornos de las tierras, extraeremos los contornos de cada una de las criaturas en nuestro campo de batalla. Una vez hayamos confirmado que el tamaño de estos contornos es muy similar al contorno de una carta, sabremos que dentro de ese contorno de la imagen tenemos una carta de criatura que debemos analizar. Una vez extraídas todas las criaturas a analizar, el proceso será el siguiente: segmentaremos la carta, quedándonos solo con la esquina inferior derecha como se puede ver en la figura 5.5, de esta manera, al aplicar el OCR obtendremos la fuerza y la resistencia de la criatura como un string en el formato Fuerza/Resistencia, que podremos separar de manera sencilla

y procesar como números. Al aplicar esto a todas las criaturas que controlamos, podremos calcular la suma de todas las fuerzas y todas las resistencias y almacenarlo en las columnas pertinentes, realizando exactamente lo mismo con las criaturas el oponente.

Figura 5.5: Ejemplo de carta de criatura en la que se ha segmentado el valor de la fuerza y resistencia, siendo Fuerza/Resistencia.

El resto de variables que nos ayudan a modelizar la situación de la partida, como son el número de cartas **exiliadas**, en el **cementerio**, en las **manos** y en la **biblioteca**, se extraen de la misma manera, por lo tanto explicaremos el proceso para todas ellas.

Para cada una deberemos extraer el segmento de la imagen en la que aparece esta, como podemos obervar en la figura 5.6. Para cada uno de estos segmentos de imagen aplicaremos un umbral con tal de diferenciar los números del fondo, y posteriormente aplicaremos el OCR, que nos devolverá el número en cuestión y que ya podremos almacenar en la columna correspondiente.

Figura 5.6: Ejemplo del número de cartas que se encuentran en juego, donde 1 correponde al número de cartas en el mazo, 2 al número de cartas en la mano, 3 al número de cartas en el cementerio, y 4 al número de cartas en el exilio.

5.3 Decisión tomada

En último lugar, estudiaremos como extraer el **log** de la partida, donde encontramos todas las deciciones que se han tomado a lo largo de la misma. Para ello es tan sencillo como irnos al log situado en el margen derecho de cualquier partida y aplicar el OCR en esta zona marcada en la figura 5.7. Tras extraer este log no se almacenará este texto directamente, si no que se pasará por un preprocesado previo que explicaremos a continuación en el procesamiento de los videos.

Figura 5.7: Ejemplo de partida de 'Magic Online' donde se ve la extracción del log (derecha, recuadro rojo).

Ahora que tenemos claro como extraer todos los campos de interés de una imagen de la partida, de forma muy simplificada solo tendremos que ir iterando en cada uno de los frames del video partida y aplicando estos métodos para extraer toda la información de la partida completa frame a frame.

Como resulta obvio, si analizamos cada uno de los frames, el resultado obtenido sería muy grande, y obtendríamos una gran cantidad de filas en el dataframe final que serían iguales. Para tratar de evitar esto, haremos uso del log. El log al final registra cualquier acción que se haya realizado, sea cual sea, por lo que al procesar la partida, si el log cambia una acción ha sido realizada y, por lo tanto, debemos analizar el frame en el que nos encontremos para registrar los cambios que haya habido en todas nuestras variables, guardando por lo tanto una nueva fila de información en nuestro dataframe.

El método más sencillo para construir este disparador que nos indique cuando se ha realizado una acción, es **comparar** el log en el frame actual con el log en el frame anterior, para ello, podemos coger ambas imágenes y restarlas. Si el resultado no es nulo, entonces significa que ambos logs son distintos y tendremos que realizar una llamada a todas nuestras funciones para capturar toda la nueva información a raíz de esta acción. En este momento, podemos recalcar, que si bien todas las variables que no son el log pueden ser distintas o iguales a pesar de la acción realizada, el log debe ser obligatoriamente distinto.

Una vez hemos terminado de procesar el video, el dataframe resultante debe ser post procesado, puesto que en los logs habrá mucha información repetida al tener solo un pequeño segmento de información adicional en comparación con el log de la jugada anterior. Al igual que con el log, los otros campos también serán post procesados para mejorar la robustez de la extracción como explicaremos en el capitulo siguiente. Una vez terminado este postprocesado ya habríamos cerrado todo el ciclo y tendriamos la información de nuestra partida ya extraída en un fichero csv que podríamos utilizar para nuestro objetivo.

CAPÍTULO 6
Mecanismos para la mejora de la robustez

En este capítulo entraremos en profundidad a analizar todos los métodos propuestos y aplicados a la hora de mejorar la robustez de nuestra herramienta, con la finalidad de construir algo que realmente funcione en múltiples escenarios reales, y no construir una herramienta poco funcional adaptada para solo algunos casos de ejemplo.

La necesidad de mejorar la robustez de nuestra herramienta nace principalmente, aunque no en su totalidad, del margen de error que el OCR tiene, ya que al depender en muchas fases de él, un error del mismo por pequeño que sea podría ser un problema en el flujo completo.

6.1 Recursos disponibles

Para solucionar los posibles problemas que puedan aparecer al utilizar el OCR para extraer de qué carta se trata, como cuando codificamos las cartas de la mano o las tierras que tenemos en juego, utilizaremos la **distancia de Levenshtein**, idea inspirada en el artículo[18]. Se trata de una solución sencilla y eficiente, puesto que como ya hemos indicado previamente, en las dos situaciones en las que queremos extraer el nombre de una carta disponemos de una lista con las cartas existentes en nuestro mazo. Esto nos permite calcular la distancia de Levenshtein de la palabra obtenida por el OCR a los nombres de las cartas en el mazo, de esta manera, el nombre de la carta que tenga un menor valor de distancia de Levenshtein con el nombre de carta extraído por el OCR será el considerado como auténtico nombre de la carta estudiada. Si bien este método nos añade robustez a la hora de detectar las cartas, puede acarrearnos también algunos problemas, puesto que si por ejemplo el OCR no detectara nada, aún así obtendríamos una carta como la más cercana (en este caso, el nombre más corto entre toda la lista de cartas), por lo que hemos considerado relevante añadir también un filtro por abajo. De esta manera si el texto extraído por el OCR es una cadena vacía o muy corta, se considera un error y no se calcula la distancia de Levenshtein. A continuación podemos ver un ejemplo de como funcionaría el flujo:

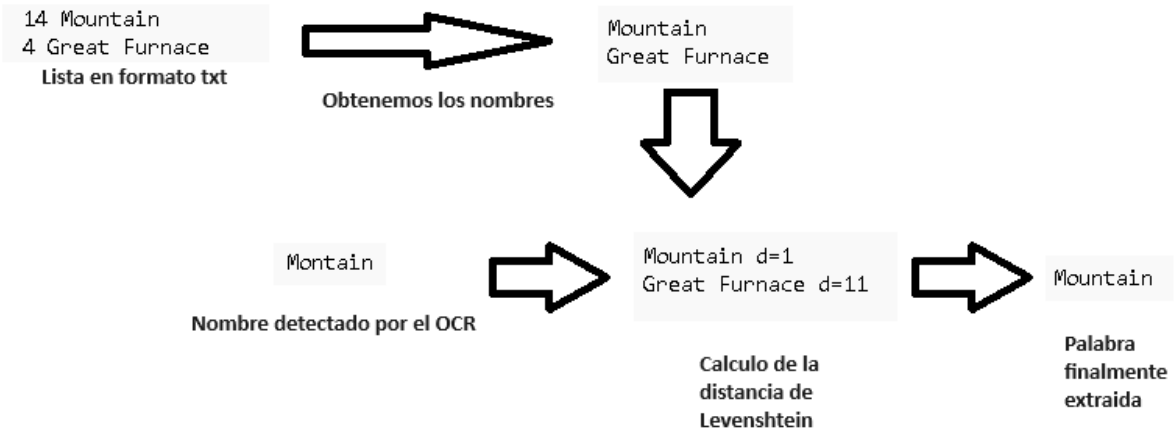

Figura 6.1: Flujo de extracción de palabras del OCR.

6.2 Estado de la partida

Una vez explicado el problema de robustez más acuciante, podemos hablar de problemas a la hora de extraer otros elementos como son los números de cartas en mano, en el exilio, en el cementerio... Para estos casos, hemos añadido una funcionalidad que se aplica post procesado del video completo. Generalmente, hay ciertas variables que deberían disminuir o aumentar dentro de un márgen concreto y el ejemplo más claro es el número de cartas en la mano. En el caso del mazo que hemos escogido para realizar las pruebas, no hay ninguna carta en el mazo que nos haga robar más de una carta, por lo que al estar capturando la información cada vez que se realiza cualquier tipo de acción, la máxima diferencia que debería haber entre el número de cartas en mano antes de que se realizara una acción cualquiera y después de que se realizara sería 1. Si por ejemplo la acción ha sido jugar una carta de la mano, entre la última vez que capturamos la información y ahora tendremos una carta menos en la mano (hemos 'perdido' la que hemos jugado). Lo mismo ocurre para el resto de variables, por lo que hemos dado la opción de tras extraer toda la información de la partida, ajustar si se desea estos márgenes para detectar si hay algún fallo a la hora de detectar algún número y reemplazarlo por su valor anterior, o un valor más realista que se desee.

Si seguimos con otros elementos, podemos tener problemas a la hora de detectar cuál es el estado del campo de batalla cuando estamos tratando de atacar o bloquear, esto se debe a que en 'Magic Online', al entrar en estas fases el campo de batalla cambia haciendo un *zoom out*, esto nos puede acarrear muchos problemas, puesto que las criaturas se desplazarán ligeramente y nuestra ventana con la que seleccionamos el campo de batalla fallará. En este caso somos capaces de detectar las cartas, pero solo fragmentos de ellas, puesto que la ventana no cubre en su totalidad las criaturas que ahora están ligeramente desplazadas. Para solucionar esta situación, hemos optado por ajusta el márgen error, de manera que cuando detectamos la criatura pero no está completa, sabemos que nos encontramos en una de estas fases especiales y debemos modificar ligeramente el controno de la carta con un margen precalculado, lo que nos permite ajustarnos a esta nueva situación y capturar la criatura ahora ya en su totalidad.

6.3 Decisión tomada

Para finalizar este apartado, hablaremos del log. Como ya se comentó en el capítulo anterior, tenemos el problema de que en el log hay una gran cantidad de información que está repetida, ya que lo único que nos resulta de interés es aquellas acciones nuevas con respecto al log anterior. La solución propuesta para este problema se aplica a modo de post procesado de los logs tras haber terminado de procesar toda la partida. Este consiste en ir iterando fila por fila, cogeremos el log de esta fila y buscarémos la cadena de texto más larga que coincida con la cadena de texto más larga posible del log anterior, y eliminaremos esa cadena de nuestro log. De esta manera, evitamos que nuestro log tenga información repetida y que sea mucho más sintético. Si bien, hay un pequeño detalle que puede tener cierto interés mencionar, y es que la extracción de los logs se realiza también mediante OCR, por lo que tenemos un cierto margen de error de letras que pueden variar de un log a otro, aunque sean exactamente el mismo texto en la realidad. Por esa razón, a la hora de comparar las cadenas para ver si son o no iguales, primero se calcula la distancia de Levenshtein, para que si las dos cadenas difieren en una cantidad menor a un umbral establecido, dichas cadenas se considerarán iguales. Esto nos permite que, si tenemos segmentos del log que sean iguales al log anterior pero que varien en alguna letra por culpa del OCR, consideremos estas cadenas iguales y podamos detectar y eliminar la información repetida a pesar del error del OCR.

En la figura 6.2 podemos ver un ejemplo de como funciona este proceso, arriba tendremos el log1 que sería el log anterior al que vamos a modificar ahora mismo, el log 2, recorreremos ambas cadenas buscando cual es la subcadena más larga que ambas comparten, pero utilizando un umbral de Levenshtein fijo, en este caso 1, para poder trabajar a pesar de algún fallo que pueda presentar el OCR, una vez detectada la subcadena, la eliminamos del segundo log y ya tenemos el resultado deseado, el segundo log ya no tiene información redundante.

Si bien puede parecer peligroso utilizar un umbral de Levenshtein para comparar las subcadenas, la realidad es distinta, puesto que cada acción nueva realizada en el log es una frase que tendrá una longitud superior sin ninguna duda a 10 caracteres, por lo que no existe un riesgo real de que se elimine valiosa información en el caso de entrar dentro del umbral y ser eliminada por error. Esto también se debe en parte a que todas las acciones tienen una estructura similar a esta: Minuto en el que se realizó la acción: Turn número del turno en el que nos encontramos Nombre del jugador acción realizada, por lo que eliminar alguna información como podría ser el minuto de la partida en el que se genera la entrada del log o el número del turno por error no serán para nada relevantes.

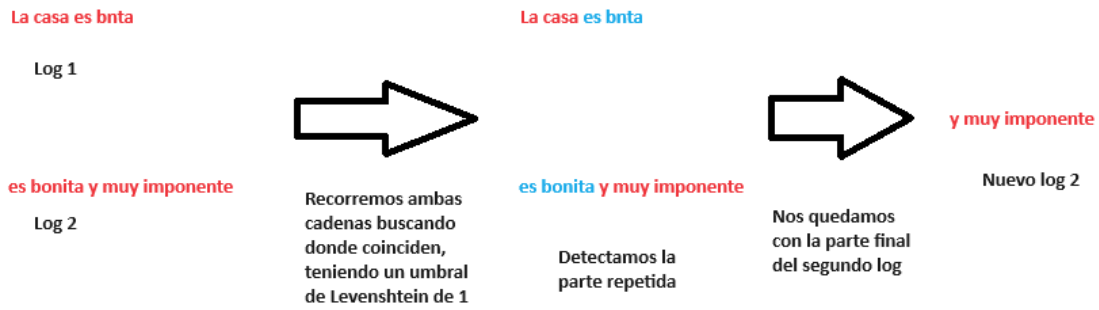

Figura 6.2: Flujo de extracción de palabras del log

CAPÍTULO 7

Conocimiento extraído y Evaluación de la solución propuesta

Ahora que ya hemos explicado como funciona nuestro mecanismo de adquisición de modelización y adquisición de información, hemos considerado de gran interés realizar unos estudios, aunque sea a pequeña escala del rendimiento, y los resultados obtenidos por parte del modelo.

Para realizar estos estudios, haremos uso de la pequeña muestra de la que disponemos, un total de treinta partidas utilizando el mismo mazo y con una duración variable de las mismas.

En primer lugar, a pesar de ser algo muy variable y que depende de la velocidad de juego de los jugadores, trataremos de hacer una pequeña aproximación de cual es la relación duración de la partida versus cantidad de acciones realizadas, para que nos podamos hacer una idea sobre cual es la cantidad de observaciones finales que extraeremos de una partida tras saber el tiempo que ha durado la misma. En la figura 7.1 podemos apreciar todas las partida con las que hemos realizado el estudio, podemos observar que existe un cierto nivel de relación como era esperable entre la duración de una partida y el número de acciones que hemos registrado, que está entorno a un 20 %, lo que significa que en el mazo concreto que hemos seleccionado para las pruebas se realiza en promedio una acción cada 20 segundos.

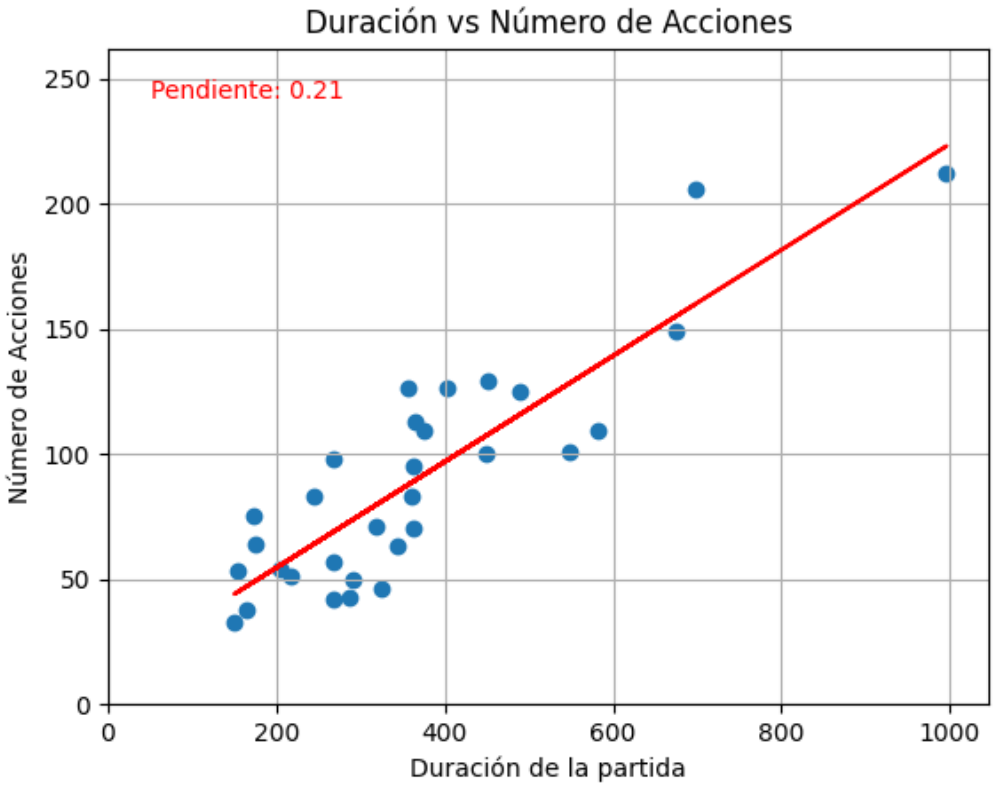

Figura 7.1: Comparativa entre la duración de la partida y el número de acciones realizadas.

En segundo lugar, hemos decidido realizar un estudio de rendimiento. Para ello, utilizaremos la pequeña muestra que hemos generado, y trataremos de preveer cuáles son las prestaciones de nuestra herramienta, estudiando la relación entre la duración de la partida y el tiempo que hemos tardado en procesar al completo la misma.

En la figura 7.2 podemos ver perfectamente la fuerte correlación que existe entre estas dos variables. Es natural que estas variables estén estrechamente relacionadas, ya que el proceso de procesamiento será igual para cada uno de los frames de un video independientemente de la duración del mismo, por lo que como podemos ver la relación indica que tardamos en procesar una partida más o menos el doble de lo que esta dura.

La poca variabilidad que podemos llegar a apreciar en el gráfico se puede deber a la necesidad por parte de la herramienta de reanalizar algunos frames debido a la utilización de alguno de los métodos de robustez, o a la situación de tener una gran cantidad de cartas en juego, lo que implicaría algo de tiempo de procesado extra, aunque probablemente nada significativo. Viendo los resultados, nos encontramos en un buen punto para poder aplicar la herramienta en tiempo real, un *delay* de un segundo por cada segundo procesado no es realmente notorio, ya que los jugadores normalmente necesitan mucho más que ese tiempo para tomar decisiones en consecuencia a una acción, por lo que podemos considerar más que aceptable el tiempo de procesado con el que estamos trabajando en caso de necesitar hacer uso de la herramienta en tiempo real.

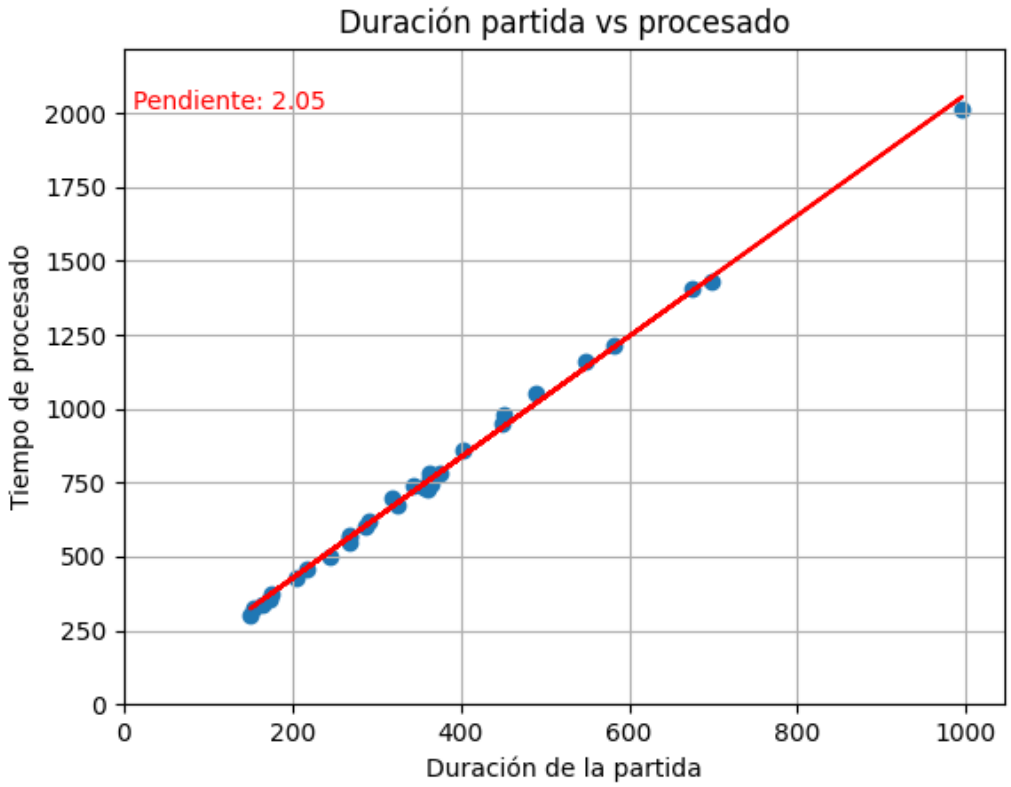

Figura 7.2: Comparativa entre la duración de la partida y el tiempo tardado en procesarla.

Si bien, podría resultar de interés estudiar cuál es el márgen de error que la herramienta puede cometer. A día de hoy imposible de caracterizar sin realizar una revisión manual de todas las partida analizadas, y aún así, encontramos una gran variabilidad entre los errores cometidos. Tras revisar a mano todas las partidas analizadas a día de hoy, se puede apreciar como la herramienta diseñada comete errores con un error cercano al 10 % de las acciones tomadas,como se puede ver en la figura 7.3 si bien este error aparece más presente en algunos campos que en otros.

Por ejemplo, campos como el log o las fuerzas y resistencias de las criaturas carecen prácticamente de errores, a su vez también carecen prácticamente de errores las codificaciones de las cartas en la mano, ya que las técnicas utilizadas son muy robustas. Por otro lado, los datos que son extraídos directamente utilizando el OCR como son el total de vidas, o variables como el número de cartas en mano, exilio, cementerio, etc., suelen concentrar una cantidad de error mayor, acercándose o sobrepasando ligeramente ese 10 %.

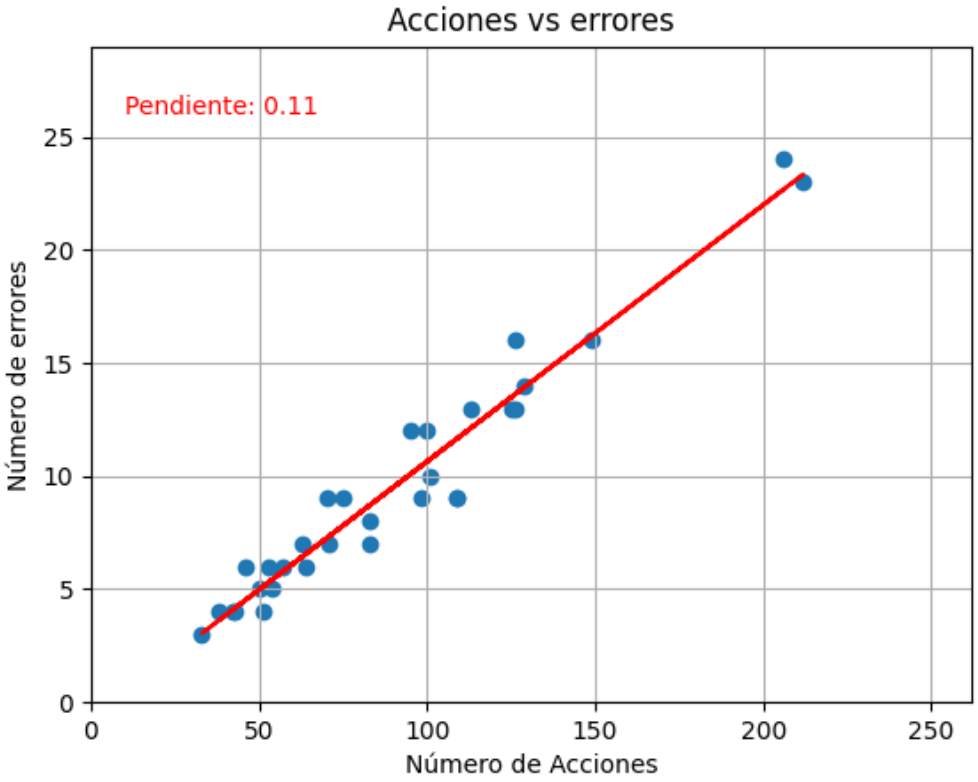

Figura 7.3: Comparativa entre el número de acciones detectas y el número de errores cometidos.

Todos estos análisis nos dan a entender que el flujo escogido es bastante bueno, y que quizás la utilización de un OCR más potente podría acabar reduciendo incluso más el márgen de error. Resulta crucial que no nos alarmemos por este error existente, ya que debemos tener en cuenta que estos errores han sido detectados antes de aplicar el post procesado, por lo que aunque el post procesado no nos asegurará que esos errores hayan sido corregidos y representen la realidad tal cual es. De este modo suaviza en gran medida el peso que puedan tener los errores, y nos permite utilizar la herramienta para extraer la información de una manera fiable y sin necesidad de estar revisando manualmente una a una las partidas.

CAPÍTULO 8
Conclusiones

Tras recorrer todos los capítulos que dan forma a esta memoria, nos encontramos ya cerca del final. Para darnos cuenta de si hemos cumplido con las espectativas planeadas al principio de este TFG, debemos dar marcha atrás y analaizar si los objetivos planteados en un principio han sido cumplidos. Para ello, iremos analizando los objetivos propuestos y hablando de qué dificultades hemos encontrado, cómo las hemos solucionado, qué errores hemos cometido y qué hemos aprendido de todo este proceso.

En primer lugar, hablaremos sobre el **primer objetivo**, que era diseñar y encontrar una forma de representar un problema con una gran cantidad de variables y complejidad. El mayor error que hemos cometido es infravalorar la magnitud del problema cuando elegimos enfrentarnos a una situación real. Sabemos que la dificultad y la complejidad que vamos a afrontar es mucho más alta que en las posibles situaciones de ejemplo o preparadas con las que hayamos aprendido. A pesar de ser conocedores del problema al que nos enfrentábamos, infravaloramos la magnitud de la tarea a realizar, teniendo que dedicar en la realidad un tiempo y esfuerzo mayor a los esperados. Esto también implicó tener que renunciar a una representación absoluta del problema, buscando aproximación lo suficientemente cercana como para aportar el valor esperado, a la vez que simplificar de alguna forma la situación de manera que fuera realista abordarla.

Si bien en este punto no eran necesarios unos conocimientos técnicos extremadamente avanzados, consideramos que esta tarea no es algo trivial que cualquiera pueda llevar a cabo, ya que es la constante necesidad de convertir el mundo que nos rodea en un problema abordable, trabajar día a día con datos, construir modelos, discernir entre variables importantes y superfluas, este constante roce, esta trivialización de algo que para mucha gente puede resultar ajeno y complejo, lo que hace que una mente obtenga un nivel de abstracción que nos permita ver el problema al que nos enfrentamos desde otro prisma, y que nos permite por lo tanto, construir una solución real y plausible del mismo.

En resumen, los errores cometidos han sido sobre todo infravalorar la magnitud del problema a enfrentar, lo que ha resultado en una necesidad de tiempo extra para pensar y replantear el enfoque que se quería tomar. Mientras que a nivel de aprendizaje ha resultado muy enriquecedor enfrentarse a una problematica de tal envergadura, confirmando que el conocimiento extraído a lo largo de toda la carrera ha servido como base para disponer de la visión y las herramientas para resolverlo, y asentando las bases del primer gran proyecto al que deberemos enfrentarnos en nuestra larga carrera.

En segundo lugar, el **objetivo técnico**, una vez habíamos ideado la forma en la que queríamos enfocar el problema, debíamos llevarlo a cabo, construyendo una herramienta que extrajera toda la información que habíamos considerado de interés. Al igual que en el punto anterior, el principal error que se ha cometido ha sido en torno a la planificación. Tras superar el primer objetivo, las sensaciones eran muy buenas, ya teníamos aborda-

do el problema de forma satisfactoria y parecía que diseñar y llevar a cabo la solución sería una tarea más sencilla. Nada más lejos de la realidad, y como suele ser costumbre, el proceso se alargó más de lo esperado. Al final, se necesitaron una gran cantidad de iteraciones para llegar a la solución final presentada en este documento, esto se debe en parte al fallo existente en las diferentes herramientas que nos planteamos utilizar.

Pongamos un ejemplo para entenderlo mejor: nosotros queremos codificar las cartas que tenemos en la mano, por lo tanto, podemos pensar que quizás un enfoque basado en reconocimiento de imagenes puede ser lo mejor. Como disponemos de las cartas que tenemos en la baraja, podemos buscar una imagen de cada una de ellas, y tratar de comparar las imagenes de diferentes formas. En su día se probaron técnicas como comparar histogramas de color, detectar formas coincidentes en una imagen en otra, utilizar redes neuronales para realizar una tarea de clasificación de imágenes, etc., pero finalmente nos decantamos por utilizar el OCR ya que era el sistema que menos margen de error tenía y más versátil era ante las diferentes variaciones que podíamos encontrar en el programa. Esto es tan solo un ejemplo, pero es obvio que esta necesidad de medir el rendimiento y probar diferentes enfoques tecnológicos para llegar a la solución puede resultar en un retraso evidente con respecto a los tiempos estipulados. Aun así, no consideraría esta situación como un error o un problema, si no algo inherente al proceso de buscar una solución, el problema aparece por la falta de experiencia a la hora de estimar la necesidad de tiempo extra que será necesaria.

A pesar de todo, estamos muy contentos con haber experimentado este proceso al completo, y por enfrentar por primera vez un proyecto de gran envergadura, en el cual debes analizar cuáles son las herramientas que tienes a tu alcance y decidir cuál crees que funciona mejor para tu problema en concreto. Si tuviéramos que centrarnos en qué conocimientos hemos adquirido de cara a cumplir este objetivo, destacaría principalmente profundizar un poco más en el procesado de imagenes y videos, aprender el uso de herramientas como el OCR utilizado en la solución final, y el estado del arte de los modelos generativos del lenguaje, para tratar de acercar la herramienta lo máximo posible a una solución útil de cara a ser utilizada en un futuro.

8.1 Legado

Como hemos hablado y visto a lo largo de todo este TFG, el tema escogido es extremadamente de conocimiento muy especñifico o nicho. Si tratamos de buscar algún tipo de proyectos similares a este, no encontraremos nada, ya que normalmente la gente que está interesada en el juego suele enfocarse más en hacer análisis más estadísticos, como los referidos a la calidad de una carta individual en función de los resultados o la cantidad de juego que está viendo, análisis en profundidad de cómo se comporta y evolucionan los mazos más jugados del juego, etc. Por lo tanto, consideramos que este trabajo puede ser de gran utilidad como base para cualquiera que en un futuro quiera analizar partidas en profundidad, realizar estudios de cara a futuro sobre el desempeño de cartas concretas en una baraja o incluso contruir modelos capaz de aprender a través de una muestra lo suficientemente grande de partidas.

En conclusión, pensamos que este trabajo puede resultar de utilidad para cualquier usuario que en un futuro desee centrarse en cualquier trabajo relacionado con partidas de 'Magic: the Gathering', además desde una doble perspectiva, ya sea leyendo y utilizando como base la forma de enfocar el problema, así como utilizando directamente la herramienta para construir los datasets que pueda necesitar para lograr sus objetivos. Por

esta razón, hemos dado de alta un GitHub [1] en el que se puede encontrar todo el código base de la herramienta, el cual puede ser directamente utilizado o servir como base y referencia sobre la que construir una herramienta de mayor alcance.

8.2 Relación del trabajo desarrollado con los estudios cursados

El desarrollo de este TFG ha sido una oportunidad para integrar y aplicar de manera práctica los conocimientos adquiridos a lo largo de la carrera en Ciencia de Datos. Este ejercicio de introspección nos permite analizar la relación que tiene este TFG con los diferentes conocimientos aprendidos a lo largo del grado. Realizaremos por lo tanto un paso por los diferentes conocimientos de los que hemos hecho uso en este trabajo.

Estadística y Probabilidad. Analizar y proponer una solucón para un problema con múltiples variables y una alta complejidad solo se puede llevar a cabo si disponemos de unos amplios conocimientos estadísticos, y estamos muy familiarizados a trabajar en este tipo de problemas.

Algoritmos y Programación. Como resulta evidente, la implementación de la herramienta final necesita de habilidades avanzadas de programación, partiendo de programación básica para ser capaces de estructurar y construir toda la base de la herramienta, hasta técnincas más concretas como el procesamiento de imagenes, videos, creación de gráficas, etc.

Machine Learning. Si bien no se ha utilizado ningún modelo de machine learning en la solución final, podemos considerar que el estudio acerca de que modelos podrían ser más prometedores en futuro, así como utilizar estructuras como redes neuronales en primeras iteraciones de la detección de imágenes, demuestran la existencia de conocimientos en machine learning adquiridos durante los estudios. A su vez, un criterio lo suficientemente sólido para seleccionar la que realmente es la mejor herramienta para nuestro problema, por encima de lanzarnos a la última tecnología a la moda. En este grado hemos aprendido infinidad de modelos y soluciones posibles, pero también que en muchas situaciones donde disponemos de una baja cantidad de muestras los métodos clásicos y básicos pueden igualar o incluso superar a los métodos más modernos.

Gestión y Análisis de Grandes Volúmenes de Datos (Big Data). Si bien de manera directa no se ha utilizado un gran volúmen de datos, queda implícito a lo largo de todo el proyecto este conocimiento, desde los mismos cimientos del trabajo, búscando crear una herramienta que sea capaz de generar esas grandes cantidades de información, hasta como enfocamos la resolución del problema y cual acaba siendo el resultado final.

Procederemos ahora a estudiar la relación del trabajo realizado a lo largo de este proyecto y las competencias transversales evaluadas a lo largo de la carrera.

Responsabilidad y Toma de Decisiones. Podemos observarla a la hora de seleccionar las herramientas adecuadas y planificar las diferentes etapas del proyecto. Las decisiones han sido tomadas en base a una experimentación continua y hemos ido seleccionando aquellas opciones que nos aportaban la mayor precisión y versatilidad, aprendiendo de los errores y mejorando continuamente, lo cual ha sido crucial para el éxito del proyecto.

Comunicación efectiva. La selección de un tema desconocido para la mayoría de la audiencia, nos ha llevado a demostrar cómo haciendo uso de la comunicación efectiva, somos capaces de transmitir opiniones y enfoques acerca de temas desconocidos para los escuchantes de manera que ellos puedan entender las razones por las cuales estas decisiones han sido tomadas.

[1] https://github.com/GitApm/tfg-repository

Innovación y creatividad. Trabajar con un tema sobre el que no hay mucha información, del que no existe una base solida, nos sirve para demostrar que podemos adaptarnos a situaciones complejas y ajenas al conocimiento de muchos, aplicando la creatividad y dando respuesta a problemas que no disponen de un estado del arte que nos indique como abordarlos.

8.3 Trabajos futuros

Cuando nos sentimos ilusionados y motivados por un proyecto en concreto muchas veces resulta extremadamente difícil encontrar cuando es el momento de parar y renunciar a que ciertas partes no van a poder realizarse en el plazo de tiempo que tenemos. Por eso consideramos que la planificación de cara a qué éramos capaces de abordar a lo largo de tiempo del que disponemos para hacer un TFG era bastante acertada.

Si bien estamos contentos con el trabajo realizado y hasta donde hemos podido llegar, nos gustaría plantear algunas cuestiones que desearíamos haber abordado y que por falta ya sea de tiempo o recursos no han sido incluidas en este TFG.

En primer lugar, nos gustaría haber podido construir un modelo de aprendizaje automático que pudieramos alimentar con los datos extraídos utilizando la herramienta para hacer recomendaciones de jugadas. Si bien tenemos claro que nos habría gustado trabajar con modelos generativos de lenguaje, que recebirían como *input* los logs anteriores al momento actual, así como los datos vectorizados de manera complementaria, e introduciendo como output el log de la jugada siguiente a los datos facilizados.

Consiguiendo de esta manera una arquitectura de modelo que recibe los datos de las diferentes acciones que han ido ocurriendo a lo largo de la partida, y nos devuleve un texto que será el log de cual debe ser la siguiente jugada. Todo esto quedaba fuera del alcance de este trabajo debido a la gran cantidad de datos necesarios para poder entrenar un modelo de dichas carácteristicas y a la falta de tiempo del que disponemos para llevar a cabo este proyecto.

En segundo lugar y a raíz de la idea anterior, nos gustaría que el uso de la herramienta se expandiera lo suficiente como para ser capaces de construir datasets muy grandes, con cientos o miles de partidas, permitiéndonos esto realizar diferentes estudios concretos con estos datos, así como alimentar los futuros modelos que cualquiera tenga la intención de construir en base a estos datos.

Por último, nos gustaría en un futuro poder explorar más opciones de tecnologías de reconocimiento de caracteres. Si bien la elegida es de las mejores versiones que podemos encontrar open source, podría ser muy interesante explorar otras opciones comerciales que tengan una mayor potencia para reducir el número de errores que podemos llegar a cometer.

Esperamos que de cara al futuro cualquier persona que quiera utilizar este trabajo como base, o quiera utilizar la tecnología para sus futuros proyectos, pueda con usarla con libertad.

Bibliografía

1. PAGÁN, B. P. LOS JUEGOS DE MESA A LO LARGO DE LAS CIVILIZACIONES. *Revista Nuclear* (2020).
2. https://es.wikipedia.org/wiki/Senet.
3. https://es.wikipedia.org/wiki/Ludus_latrunculorum.
4. https://es.wikipedia.org/wiki/Richard_Garfield.
5. SHANNON, C. E. Programming a Computer for Playing Chess. *Philosophical Magazine* **41** (1950).
6. Austin Herrick, A. S. Magic: The Gathering is Turing Complete (2019).
7. https://en.wikipedia.org/wiki/Halting_problem.
8. Ward, C. D. y Cowling, P. I. *Monte Carlo search applied to card selection in Magic: The Gathering* en *2009 IEEE Symposium on Computational Intelligence and Games* (2009).
9. Švelch, J. Mediatization of a card game: Magic: The Gathering, esports, and streaming. *Sage Journals* **42** (2019).
10. KURTZ, GABRIELA, F. M. *Conservadorismo e masculinidade tóxica na cultura gamer: uma aproximação a Magic: The Gathering*. https://www.redalyc.org/articulo.oa?id=143068488020.
11. Fludal, Knut Aron, B. *Deckbuilding in Magic: The Gathering Using a Genetic Algorithm* http://hdl.handle.net/11250/2462429.
12. Arseny S. Khakhalin, H. B. D. D. *AI solutions for drafting in Magic: the Gathering* en *2021 IEEE Conference on Games (CoG)* (2021).
13. Lukasz Grela, S. S. C. Gym-RTS: Toward Affordable Full Game Real-time Strategy Games Research with Deep Reinforcement Learning. *arXiv:2105.13807*. https://arxiv.org/abs/2105.13807 (2021).
14. Zha, D. *et al.* RLCard: A Toolkit for Reinforcement Learning in Card Games. *CoRR* **abs/1910.04376.** arXiv: 1910.04376. http://arxiv.org/abs/1910.04376 (2019).
15. Mao, S. *et al. ALYMPICS: LLM Agents Meet Game Theory – Exploring Strategic Decision-Making with AI Agents* 2024. arXiv: 2311.03220 [cs.CL].
16. Niklaus, J., Alberti, M., Ingold, R., Stolze, M. y Koller, T. *Challenging Human Supremacy: Evaluating Monte Carlo Tree Search and Deep Learning for the Trick Taking Card Game Jass* en *Artificial Intelligence and Soft Computing* (eds. Rutkowski, L. *et al.*) (Springer International Publishing, Cham, 2020), 505-517. ISBN: 978-3-030-61534-5.
17. Smith, R. *An Overview of the Tesseract OCR Engine* en *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)* **2** (2007).
18. Po, D. K. Similarity Based Information Retrieval Using Levenshtein Distance Algorithm. *International Journal of Advances in Scientific Research and Engineering (ijasre)* **6** (2020).

APÉNDICE A
Objetivos de desarrollo sostenible

Grado de relación con los Objetivos de Desarrollo Sostenible (ODS).

Objetivos de Desarrollo Sostenible	**Alto**	**Medio**	**Bajo**	**No procede**
ODS1. Fin de la pobreza				**X**
ODS2. Hambre cero				**X**
ODS3. Salud y bienestar				**X**
ODS4. Educación de calidad		**X**		
ODS5. Igualdad de género		**X**		
ODS6. Agua limpia y saneamiento				**X**
ODS7. Energía asequible y no contaminable				**X**
ODS8. Trabajo decente y crecimiento económico				**X**
ODS9. Industria, innovación e infraestructuras		**X**		
ODS10. Reducción de las desigualdades		**X**		
ODS11. Ciudades y comunidades sostenibles				**X**
ODS12. Producción y consumo responsables				**X**
ODS13. Acción por el clima				**X**
ODS14. Vida submarina				**X**
ODS15. Vida de ecosistemas terrestres				**X**
ODS16. Paz, justicia e instituciones sólidas				**X**
ODS17. Alianzas para lograr objetivos				**X**

Reflexión sobre la relación del TFG con los ODS más relacionados.

ODS4.Educación de calidad. La propia existencia de este TFG es una demostración de la calidad de la educación impartida. No solo se demuestran una gran cantidad de habilidades técnicas avanzadas, sino que se muestra cómo, a partir de una base de conocimientos, estos pueden ser aplicados a temas diversos y que puedan suscitar un interés mayor para los alumnos, gracias a la flexibilidad y la diversidad de conocimientos aprendidos. Además, la existencia de este trabajo puede fomentar el aprendizaje y la investigación en todo tipo de campos diversos, sirviendo como ejemplo de que todo conocimiento se puede aplicar en un campo que pueda ser de nuestro interés, beneficiando a aquellos futuros estudiantes que quieran utilizar este trabajo como base para estudios más avanzados o aplicaciones prácticas.

ODS9.Industria, innovación e infraestructuras. La relación con este ODS es principalmente desde la parte de innovación, puesto que, como hemos podido apreciar a lo largo de todo el trabajo, al fin y al cabo, el problema al que nos enfrentamos es algo inusual y que todavía no había sido tratado. Por lo tanto, ha sido necesaria una gran cantidad de nuevas ideas, tecnologías e innovación.

ODS10.Reducción de las desigualdades. ODS5.Igualdad de género Si nos detenemos a estudiar la población que puede estar interesada en un juego como es 'Magic: The Gathering', nos daremos cuenta de que en los últimos años ha crecido y se encuentra en auge. Hace algunos años se interesaban por el juego principalmente hombres de mediana edad, mientras que a día de hoy, gracias a múltiples campañas de marketing y a la normalización de este tipo de hobbies, el juego está llamando la atención de personas de todo tipo. Con el uso de la herramienta se podría llegar a reducir la brecha de conocimiento que puede existir entre personas novatas, con menos conocimiento, personas con menos tiempo libre que dedicar al juego, y aquellas personas que se dedican profesionalmente al mismo. Sirviendo como puente para que toda persona que lo desee, independientemente de su condición, pueda equipararse a cualquier otro jugador, construyendo así un entorno competitivo con un nivel mucho más equilibrado y que mantenga el interés de todo tipo de jugadores.