UNIVERSIDAD AUTÓNOMA DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

TRABAJO FIN DE MÁSTER

Predicción de conducta humana a través de Aprendizaje por Refuerzo Profundo

Máster Universitario en Ingeniería de Telecomunicación

Autor: Alejandro Romero del Campo
Tutor: Aythami Morales Moreno
FECHA: Junio 2021

PREDICCIÓN DE CONDUCTA HUMANA A TRAVÉS DE APRENDIZAJE POR REFUERZO PROFUNDO

AUTOR: Alejandro Romero del Campo
DIRECTOR: Aythami Morales Moreno

Dpto. de Tecnología electrónica y de las comunicaciones
Escuela Politcnica Superior
Universidad Autnoma de Madrid
Junio 2021

Resumen

Resumen

El objetivo principal de este trabajo de fin de máster es la predicción e identificación de patrones en la conducta humana, mediante la aplicación de algoritmos de aprendizaje por refuerzo. Para ello se ha desarrollado un agente capaz de aprender en base a un entorno artificialmente generado, en el cual, este agente, se basa en las características visuales de rostros humanos para encontrar estos patrones. Una vez se han encontrado estos patrones, el objetivo del agente es aprovecharlos para, en un juego de decisión en el que se enfrenta a la persona de cuya imagen dispone, obtener el mayor beneficio posible a largo plazo.

Este agente, mediante un algoritmo de aprendizaje por refuerzo denominado *Q-Learning* y redes neuronales, será capaz de desarrollar un proceso de aprendizaje a lo largo de las iteraciones.

A su vez, se ha desarrollado un sistema de generación del entorno (la información a la que accede el agente para poder aprender) basado en imágenes. Estas imágenes son rostros de personas procedentes de una base de datos generada artificialmente, de cara a respetar la privacidad.

Adicionalmente al agente y al entorno que le rodeará, se han desarrollado varios sistemas de recompensas que el agente tomará como referencia para poder aprender. Estos sistemas recompensarán al agente positivamente cuando realice determinadas acciones y negativamente al contrario. El sistema de recompensa final que se ha desarrollado corresponde a una similitud con el popular dilema del prisionero, en el cual dos personas son recompensadas positivamente a largo plazo si colaboran y negativamente si ambas deciden no hacerlo.

Una vez se han desarrollado los elementos principales para ejecutar el algoritmo de aprendizaje por refuerzo, se realizará una serie de pruebas y experimentos para evaluar su rendimiento. En estas pruebas se modificarán características del agente, el sistema de recompensa o se modificará, artificialmente, el entorno para crear patrones de comportamiento y observar si un agente, con las suficientes iteraciones, es capaz de detectar estos patrones de conducta.

Para cada prueba se describirá el contexto de la misma y se analizarán sus resultados individualmente.

Finalmente, tras todas las pruebas realizadas, se enunciarán unas conclusiones globales y se describirá que trabajo futuro se podría realizar en base a la temática y resultados de este trabajo de fin de máster.

Palabras Clave

Aprendizaje por refuerzo, agente, entorno, recompensa, iteración, episodio, acción, decisión, modelo facial, patrón, comportamiento, conducta, sintética, algoritmo

Abstract

The main objective of this master's thesis is the prediction and identification of patterns in human behavior through the application of reinforcement learning algorithms. For this purpose, an agent capable of learning based on an artificially generated environment has been developed, in which this agent is based on the visual characteristics of human faces to find these patterns. Once these patterns have been found, the agent's objective is to take advantage of them in a decision game in which it faces the person whose image it has, in order to obtain the greatest possible benefit in the long term.

This agent, with a reinforcement learning algorithm called *Q-Learning* and neural networks, will be able to develop a learning process throughout the iterations.

Additionaly, an environment generation system has been developed (the information accessed by the agent in order to learn) based on images. These images are faces of people from an artificially generated database, in order to respect privacy.

In addition to the agent and the environment that will surround it, several reward systems have been developed that the agent will take as a reference in order to learn. These systems will reward the agent positively when it performs certain actions and negatively on the contrary. The final reward system that has been developed corresponds to a similarity with the popular prisoner's dilemma, in which two people are rewarded positively in the long run if they collaborate and negatively if both decide not to.

Once the main elements for running the reinforcement learning algorithm have been developed, a series of tests and experiments will be conducted to evaluate its performance. In these tests we will modify features of the agent, the reward system or artificially modify the environment to create patterns of behavior and observe whether an agent, with enough iterations, is able to detect these patterns of behavior.

For each test, the context of the test will be described and its results will be analyzed individually.

Finally, after all the tests performed, some global conclusions will be stated and future work will be described based on the subject and results of this master's thesis.

Key words

Reinforcement learning, agent, environment, reward, iteration, episode, action, decision, facial model, pattern, behavior,conduct, behavioral, synthetic, algorithm

Agradecimientos

En primer lugar, agradecer a mi tutor Aythami Morales Moreno, por toda la dedicación que ha tenido conmigo, ayudándome en todos los problemas que han podido surgir realizando este trabajo y guiándome en todo momento para alcanzar los mejores resultados posibles y que este trabajo sea posible.

En segundo lugar, agradecer a mi familia y amigos, por apoyarme en todo momento durante toda mi etapa universitaria. Por que está claro que sin ellos no habría podido llegar a donde estoy.

Por último, y no menos importante, agradecer el apoyo a Cristina. Por no cansarse de escucharme hablar de mi trabajo y, aunque este año no han sido posibles tantas horas de biblioteca, por animarme en todo momento a seguir avanzando.

Índice general

Índice de figuras

1 Introducción

1.1. Motivación del proyecto

Durante los últimos años, el campo de la inteligencia artificial ha adquirido cada vez más popularidad con multitud de líneas de investigación asociadas y grandes avances en diferentes campos de aplicación. Estos avances han sido posibles gracias al desarrollo tecnológico de los últimos años, que ha tenido como grandes protagonistas algoritmos como las redes neuronales profundas o, el tema a tratar en este trabajo de fin de máster, el aprendizaje por refuerzo.

La historia que engloba al aprendizaje por refuerzo, como ocurre con las técnicas de inteligencia artificial más modernas, es muy reciente, con avances de gran importancia durante estos últimos años e incluso en meses recientes, por lo que es un campo en constante actualización. [5]

El aprendizaje por refuerzo se basa en la creación de un agente que, dado un entorno (virtualmente creado o dado) aprende del mismo para maximizar la recompensa obtenida de un sistema que el usuario como programador ha definido previamente. El punto clave de esta técnica es la poca supervisión humana necesaria durante el proceso de aprendizaje, en comparación de otras técnicas.

Tanto el **agente**, como el **entorno** y la **recompensa**, son los tres bloques fundamentales de todo algoritmo de aprendizaje por refuerzo. Del funcionamiento en conjunto de estos tres dependerán los resultados obtenidos en el futuro.

El agente es el módulo principal del algoritmo. Este módulo será el encargado de tomar las decisiones y que llevará a cabo el proceso de aprendizaje necesario para que estas decisiones que toma sean las más óptimas posibles. Dentro de el mismo se encuentra una red neuronal que hará posible este proceso de aprendizaje.

En cuanto al entorno, ya mencionado, es todo aquello en lo que se basa el agente para tomar sus decisiones y poder aprender. Para compararlo con el mundo real, sería lo que el agente observar, si el agente fuera un ser humano, sería todo lo que ve por sus ojos. En el caso de este trabajo, se describirán experimentos en los que el entorno es una imagen y, por lo tanto, el agente se basa en cada uno de los parámetros de esa imagen para tomar sus decisiones.

Por último, el sistema de recompensas, permite definir unas reglas básicas y cerradas en relación al entorno y al proceso de aprendizaje del agente. El sistema de recompensas tiene una gran importancia, ya que de el dependerán las decisiones que tome el agente. Por este módulo

es por lo que, en muchos casos, los algoritmos de aprendizaje por refuerzo son desarrollados en juegos, ya sea juegos de mesa tradicionales o videojuegos actualmente. Y es por que, en todos estos juegos, existe un sistema de recompensas cerrado el cual no puede ser ignorado ni nadie puede estar por encima de él. En el mundo real, esto es mucho más complejo, ya que no existe un sistema de recompensas cerrado del todo, siempre pueden existir matices y cuanto más cerrado sea el sistema, más óptimo será el proceso de aprendizaje de un agente.

Una vez se conocen lo módulos principales, el objetivo de un algoritmo de aprendizaje por refuerzo está claro. El objetivo de estos algoritmos es maximizar la recompensa del agente a través del conocimiento extraído del entorno y las recompensas pasadas.[3]

En este TFM se ha partido de numerosos ejemplos y casos de estudio como son los populares desarrollos de *Deepmind* y David Silver[2] y se han tomado como motivación inicial para la creación de un entorno y un agente propio, el cual aprende a partir de un modelado biométrico de la información y sea capaz de encontrar patrones en la conducta humana, a partir de estos rasgos biométricos.

En el caso de este trabajo, se pretende plantear un entorno en el cual aprender a través de modelado biométrico de la información. En el entorno propuesto, el agente recibirá como *input* la cara de un contrincante contra el que jugará, por lo que necesitará, además de los algoritmos de aprendizaje por refuerzo, la capacidad de procesamiento biométrico para, en función de ellas, tomar decisiones. Se trata de un juego de decisión basado en el popular dilema del prisionero en la que el agente erigirá un valor, independientemente de su contrincante, y que, una vez hayan decidido los dos, en función de sus decisiones se recibirá una recompensa positiva o negativa.

1.2. Objetivos y enfoque

El objetivo principal de este trabajo de fin de máster es desarrollar un entorno experimental en el cual poder estudiar y desarrollar algoritmos de aprendizaje por refuerzo en aplicaciones relacionadas con el análisis facial. Con este fin se propone desarrollar un agente que, con un entorno previamente definido y establecido, sea capaz de encontrar patrones en el comportamiento, según las características visuales de la persona que tenga como oponente, en un juego sencillo de predicción del comportamiento humano desarrollado. De esta forma se podrán observar que rasgos le son más importantes al agente para alcanzar los mejores resultados posibles, todo este proceso, sin interacción humana de por medio.

Previamente a la creación del agente, el objetivo también es el crear un entorno que se base en la utilización de una representación aprendida de la información facial (*embedding* facial) de fotografías (es la información que verá el agente para realizar su aprendizaje, como si tuviera la cara de su contrincante delante) y un sistema de recompensas, tanto negativas como positivas, que creen un estímulo en el agente para que pueda desarrollar su aprendizaje de manera correcta.

Por último, una vez se han creado tanto agente como entorno y se ha hecho un estudio exhaustivo de como funcionan los algoritmos de aprendizaje por refuerzo, así como una comparación con otros métodos de aprendizaje, se desarrollarán los objetivos finales. Estos objetivos son la realización de múltiples pruebas variando los parámetros del entorno, del agente y del sistema de recompensas para alcanzar los mejores resultados posibles y su posterior análisis y representación con el fin de entender el comportamiento de un agente en un entorno como el desarrollado en este trabajo

1.3. Metodología y plan de trabajo

A continuación, a modo de lista, queda representado, todo el proceso de realización del presente trabajo de fin de máster.

- Documentación previa y análisis 01/07/2020 - 01/10/2020
  - Bibliografía recomendada 01/07/2020 - 03/08/2020
  - Estudio de los algoritmos dentro del aprendizaje por refuerzo 03/08/2020 - 01/09/2020
  - Prueba de diferentes entornos ya programados 01/09/2020 - 01/10/2020
- Diseño y desarrollo 01/10/2020 - 01/04/2021
  - Desarrollo del agente 01/10/2020 - 01/12/2020
  - Desarrollo del sistema de generación del entorno 01/10/2020 - 01/12/2020
  - Desarrollo del sistema de recompensas 01/10/2020 - 01/12/2020
  - Ajustes de los sistemas según las pruebas realizadas 01/12/2020 - 01/04/2021
- Pruebas y validación 01/12/2020 - 01/06/2021
  - Pruebas variando los parámetros del agente 01/12/2020 - 01/02/2021
  - Pruebas variando el sistema de generación del entorno 01/01/2021 - 01/04/2021
  - Pruebas variando el sistema de recompensas 01/03/2021 - 01/04/2021
  - Análisis de resultados y comparación con otras técnicas 14/04/2021 - 01/06/2021
- Elaboración de la memoria 14/04/2021 - 18/06/2021

1.4. Organización de la memoria

Durante esta sección se va a describir como se han estructurado las diferentes partes de la memoria y que información contiene cada una de ellas.

Como se ha podido comprobar, el primer capítulo corresponde a la introducción. En este capítulo se ha realizado un primer contacto con los términos fundamentales del aprendizaje por refuerzo y su funcionamiento. También se han expuesto cuáles serán los objetivos de este trabajo de fin de máster, algo que se ha tenido presente durante todo el trabajo y que, en el capítulo de conclusiones, se repasará si se han alcanzado dichos objetivos o cuáles de los mismos.

El siguiente capítulo es el estado del arte. En el mismo se realizará un repaso por las bases fundamentales de los algoritmos y métodos existentes dentro del propio aprendizaje por refuerzo y se profundizará en dos casos de uso que han servido de inspiración para este trabajo como lo son el algoritmo de *AlphaGo* o la empresa de *OpenAI*.

En el tercer capítulo se realizará todo el desarrollo previo a la realización de los experimentos. Primero se realizará una breve introducción acerca del funcionamiento del aprendizaje por refuerzo (sus términos fundamentales y métodos diferentes se han descrito anteriormente). A continuación se describirá el proceso del desarrollo del agente, del sistema generador del entorno, junto con el modelo facial que extrae información a partir de las imágenes de la base de datos (describiendo también esta base de datos utilizada).

A continuación, el siguiente capítulo se centrará en los experimentos realizados. Se describirá cada uno de ellos, exponiendo primero las condiciones y que se pretende alcanzar, y finalmente

se expondrán sus resultados, analizándolos. Este capítulo, junto con el anterior, corresponde a la mayor parte del trabajo realizado.

Para finalizar, el capítulo de conclusiones repasará los objetivos iniciales del trabajo y qué se ha podido alcanzar en la realización del mismo. También se analizará cual podría ser el trabajo futuro a realizar, en una hipotética continuación de este trabajo.

2
Estado del Arte

2.1. Introducción

En esta sección se expondrá el estado del arte acerca de los diferentes algoritmos y técnicas que han supuesto avances en el tema principal de este trabajo de fin de máster, el aprendizaje por refuerzo. Se hará un repaso por los autores y organizaciones más importantes en la historia de esta técnica de inteligencia artificial.

La historia que engloba al aprendizaje por refuerzo, como ocurre con las técnicas de inteligencia artificial más modernas, es muy reciente, con avances de gran importancia durante estos años e incluso en meses recientes, por lo que es un campo en constante actualización, algo que se ha tenido en cuenta a la hora de estudiar el estado del arte.

Por último, se va a estudiar una de las plataformas actuales que sirven para observar comportamientos de agentes y entornos de aprendizaje por refuerzo. Esta plataforma tiene una gran popularidad hoy en día y prácticamente cualquier proyecto dentro del aprendizaje por refuerzo ha hecho uso de ella ya sea como un paso inicial (como es el caso de este trabajo) o como entorno para probar agentes y algoritmos propios.

2.2. Aprendizaje por refuerzo

Durante esta sección se describirá el estado de arte actual que engloba a todos lo algoritmos de aprendizaje por refuerzo. En secciones posteriores se explicarán algoritmos en concreto y casos que han servido de inspiración para la elaboración de este trabajo. Durante esta sección no se describirá el funcionamiento como tal de los principios básicos del aprendizaje por refuerzo, es algo que se ha descrito brevemente en la introducción y que, en el capítulo que describe los experimentos realizados, se realizará también.

Como se ha comentado anteriormente, el aprendizaje por refuerzo, al igual que otros algoritmos de inteligencia artificial conocidos, es un área muy dinámica, con una constante evolución, especialmente en los últimos años.

Primero se va a describir en qué se diferencia un algoritmo de aprendizaje por refuerzo de otros algoritmos de aprendizaje automático. En un algoritmo supervisado, existe una señal de

error, conocida como gradiente, mediante la cual se pueden optimizar los resultados. En cambio, en algoritmos no supervisados, como es el caso de clustering, no existe esta señal de error, por lo que se evalúa el modelo completo. Un algoritmo de aprendizaje por refuerzo se puede dividir en dos partes fundamentales del proceso de optimización. Una primera parte de prueba y error en la cual, a base de fuerza bruta, se prueban resultados y una parte de memoria, en la cual, los mejores resultados obtenidos son guardados para, en siguientes iteraciones, partir de esos puntos.

El concepto de aprendizaje por refuerzo tiene su origen en la psicología, más concretamente en la rama que estudia la conducta humana y el comportamiento. Uno de los estudios más importantes en este campo y que sirvieron de influencia para el aprendizaje por refuerzo fue el realizado por el psicólogo J. B. Watson (1878-1958)[6]. J. B. Watson afirmaba en sus estudios que, para modificar una conducta se necesita un estímulo, es decir, una recompensa, ya sea negativa o positiva. Como se va a observar, esto es algo que tiene gran influencia en el aprendizaje por refuerzo. El agente, a través de este estímulo (recompensa) optimiza sus resultados (modifica su conducta).

No fue hasta la década de los 60-70 cuando se realizaron los primeros estudios de inteligencia artificial que incorporaban el término de aprendizaje por refuerzo. Los primeros acercamientos a algoritmos de aprendizaje por refuerzo utilizaban métodos para resolver los problemas basados en prueba y error, llegando a los mejores resultados posibles mediante fuerza bruta y un gran número de iteraciones. Aunque pueda parecer primitivo, este método no es muy distante a lo utilizado hoy en día.

A día de hoy existen tres métodos fundamentales dentro del aprendizaje por refuerzo:

- Programación dinámica y control óptimo: Un método que surgió como alternativa a los métodos de prueba y error. Se basa en alcanzar los resultados más óptimos posibles reduciendo el tiempo de ejecución al máximo. Tiene su orinen en la carrera espacial y se basa en dividir el problema en problemas reducido. El problema es que se basa, también, en que, para cualquier estado, el entorno va a ser siempre el mismo, algo que en la mayoría de experimentos de aprendizaje por refuerzo no ocurre, es por esto por lo que no es el método más utilizado.
- Método de Monte Carlo: Se basa en la utilización de procesos de decisión de Markov, en los cuales no se toman decisiones en base a estados pasados y el entorno, visto por el agente, no varía. Es utilizado en algunos estudios de aprendizaje por refuerzo pero, en el caso de este trabajo, no será el método utilizado.
- Métodos basados en diferencias temporales: Método en el cual los resultados pasados, así como la recompensa acumulada de iteraciones anteriores cobra una gran importancia e influye en los resultados del estado actual. El algoritmo más utilizado dento de este método es el *Q-Learning*.

En el caso de este trabajo, se va a utilizar un algoritmo perteneciente al método de diferencias temporales, más concretamente el popular algoritmo de Q-Learning. Durante la descripción del agente, en la sección 3.5, se desarrollará más acerca de este método. Como se ha mencionado en algorimto *Q-Learning* es uno de los más populares dentro del aprendizaje por refuerzo. En concreto se realizará un versión denominada *Deep Q-Learning* en la que se incorporan redes neuronales profundas a este proceso de aprendizaje.

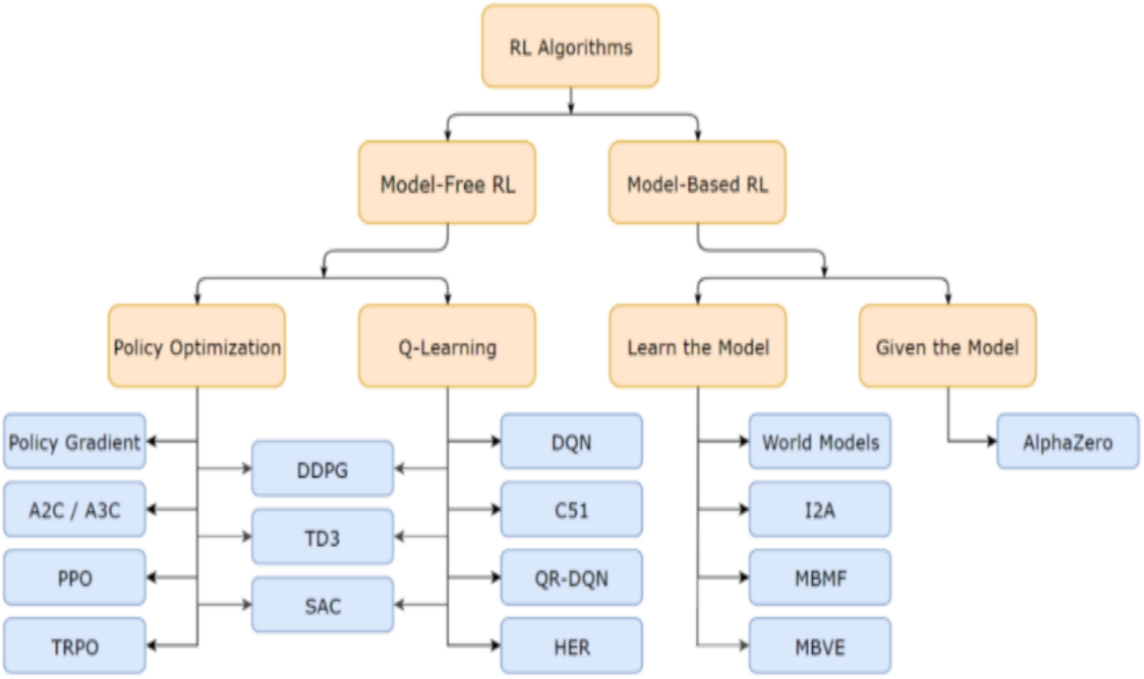


Figura 2.1: Algorimtos dentro del Aprendizaje por Refuerzo [1]

En la figura 2.1 se puede observar un esquema con los diferentes algoritmos dentro del aprendizaje por refuerzo, según su categoría principal. Como se puede observar, el algoritmo de *Q-Learning* pertenece a aquellos que no están basados en un modelo. Esto quiere decir, que son algoritmos basados en los procesos de prueba y error. Por simplificar, el agente, dentro de algoritmo basados en modelos, puede predecir cual va a ser el estado y la recompensa futura, en los que no están basados en modelos no. Dentro de los algoritmos de *Q-Learning* existen variaciones, como la del caso de este trabajo *DQN*.

En la siguiente sección se describirá uno de los casos de uso más popular dentro del aprendizaje por refuerzo, aplicado a juegos, que ha servido de gran influencia para la realización de este trabajo de fin de máster.

2.3. Sobre el caso AlphaGo

A lo largo de estos últimos años, numerosos algoritmos han alcanzado popularidad por vencer a jugadores expertos en diferentes campos como el ajedrez, Go y el póker. Todos estos algoritmos están relacionados con el tema principal de este trabajo de fin de máster, el aprendizaje por refuerzo.[2]

Para hablar del estado de arte del aprendizaje por refuerzo es necesario hablar de *AlphaGo*.

La figura detrás de *AlphaGo*, sus algoritmos y sistemas es David silver. Alrededor del año 2014 fue cuando David comenzó sus investigaciones para desarrollar su primer sistema, *AlphaGo*, un sistema diseñado íntegramente para competir en el juego del Go mediante aprendizaje por refuerzo. David silver se encontraba realizando trabajos de investigación en la universidad

Este sistema supuso un gran avance respecto a otros sistemas ya existente. *AlphaGo* ejecutándose en una sola máquina logró ganar 50 partidas a estos sistemas excepto una. Pero en

una partida igualada (*AlphaGo* ejecutándose en varias máquinas, al igual que sus contrincantes) ganó las 500 partidas.

Pero no fue hasta el año 2016 cuando ocurrió el acontecimiento que lo cambió todo. El sistema *AlphaGo* ganó al 18 veces al campeón del mundo Lee Sedll por 4-1 (al mejor de 5) por un premio de 1 millón de dólares que fue donado en un acto significativo.

El hecho de que Lee ganara, aunque fuera una sola vez, a *AlphaGo* es de gran importancia, ya que es la única persona capaz de ganar al sistema *AlphaGo*, no sólo respecto al Go, si no a sus 74 juegos y variantes.

Dos años después, en 2017, *DeepMind*, la empresa encargada de *AlphaGo* entre otros, desarrolló una nueva versión de su sistema, *AlphaGo Zero* que consiguió ganar al anterior sistema, con un increíble 100-0. Todo este proceso de aprendizaje es por lo que es considerado aprendizaje por refuerzo, por que se llegó a este increíble avance mediante la repetición de partidas y las horas de entrenamiento desde un estado inicial básico hasta encontrar estrategias más propias de un "superhumano". Todo esto sin ninguna base de datos ni interacción humana, solo conociendo las reglas del juego, algo que se busca en este trabajo y que ha supuesto un gran avance en el ámbito de la inteligencia artificial, sugiriendo incluso que la inteligencia artificial, en un futuro, no necesitará de personas para alcanzar resultados dignos de "superhumanos".

Un poco más adelante, dentro del mismo año, *DeepMind*, anunció su nueva línea de estudio, al crear lo que sería su siguiente sistema y el que actualmente se sigue desarrollando, *AlphaZero*. Su nombre ya lo indica todo, se desliga totalmente del juego de Go y su nuevo objetivo es la creación de un algoritmo que se pueda aplicar en general para aprender a realizar una tarea sin la intervención de humano en todo ese proceso de aprendizaje.

En la figura 2.2 se puede observar el proceso de aprendizaje del sistema de *AlphaZero*. Se puede apreciar como, al cabo de varios días de entrenamiento, se superan los resultados de otros sistemas anteriores. Los resultados que se observarán en los experimentos realizados en este trabajo seguirán un formato similar a este, con una gráfica con la evolución del aprendizaje, en función del tiempo de entrenamiento. Algo que, sin duda, llama la atención, es la cantidad de tiempo de entrenamiento empleado, algo que es necesario para alcanzar los mejores resultados posibles.

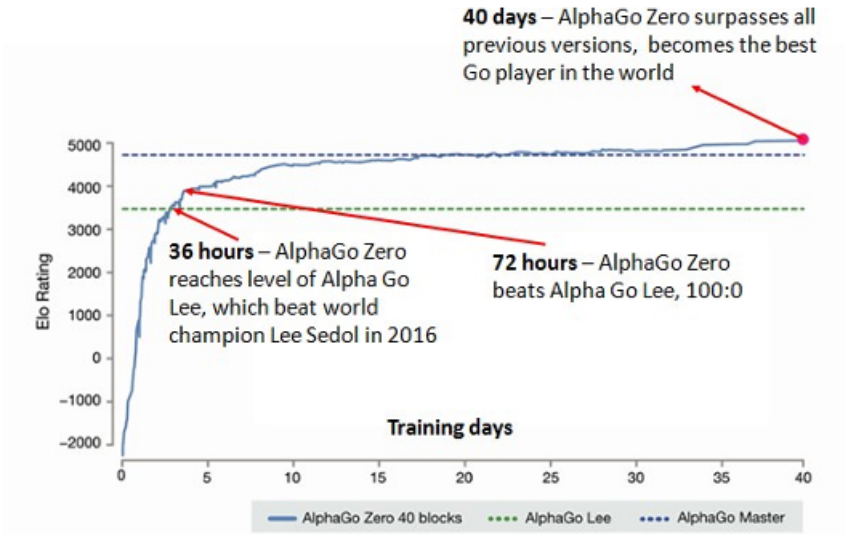


Figura 2.2: Proceso de entrenamiento de *AlphaGo Zero*.[2]

Como se ha comentado anteriormente, existen avances en los meses recientes en el campo del aprendizaje por refuerzo, y esto es en referencia al nuevo articulo publicado en la revista *Nature* por David Silver en diciembre de 2020, sobre el nuevo algoritmo *MuZero*.[2]

Este nuevo algoritmo ha sido probado, en primer lugar, con juegos de la Atari 2600 y se alcanzó el rendimiento del estado del arte. Después fue probado con juegos mas complejos como el Go o el ajedrez y, sin conocimiento de las dinámicas del juego, logró alcanzar los resultados, propios de un "superhumano", alcanzados por el algoritmo de *AlphaZero* y anteriores, que, a pesar de que no requerían de una interacción humana, si que necesitaban de conocer la dinámica (reglas básicas) del juego, previamente de comenzar el aprendizaje.

El no necesitar de ninguna regla de juego, hacen que se puedan aplicar a problemas del tiempo real. Este es uno de los problemas del aprendizaje por refuerzo al aplicar sus algoritmos a problemas de la vida real y, a la vez, una de sus virtudes al aplicarlo en entornos como juegos.

En un juego, ya existen unas reglas definidas, más o menos complejas, pero en las cuales el jugador debe acatarlas y no puede saltárselas ni pueden darse situaciones fuera de ellas. Este entorno tan definido, para un algoritmo, es algo extremadamente positivo, mientras que, si queremos aplicarlo a problemas reales, en la vida no existen unas reglas tan definidas y cerradas. Por ello, avances como MuZero son determinantes y muy positivos para poder llevar el aprendizaje por refuerzo a cualquier problema real.

2.4. *OpenAI* y el entorno *Gym*

Adicionalemnete a todo lo logrado con *DeepMind* y *AlphaGo* también es necesario nombrar a *OpenAI* ya que ha influido significativamente en este trabajo.

OpenAI es una compañía de inteligencia artificial centrada íntegramente en labores de investigación sin ánimo de lucro fundada por el archiconocido Elon Musk.

Recientemente esta compañía ha adquirido mucha popularidad debido a su último proyecto, la tercer versión de GPT, GPT-3[7], su modelo de lenguaje predictivo que ha cedido su exclusividad a Microsoft. Mediante este modelo se han observado resultados espectaculares teniendo incluso una inteligencia artificial con capacidad de crear código de programación a partir de simples frases en lenguaje natural.

Pero durante la realización de este trabajo se ha obtenido la inspiración de su proyecto *Gym*[8]. *Gym* es una plataforma de investigación para el aprendizaje por refuerzo (tema central de este trabajo) que incorpora librerías muy interesantes y que fue lanzado en 2016.

Gym nos permite utilizar un gran numero de entornos predefinidos para probar la eficacia de nuestros agentes. Estos entornos componen un gran número de juegos clásicos conocidos y que son sencillos de instalar y trabajar con ellos. Actualmente sus librerías están disponibles para el lenguaje Python, algo muy conveniente para este trabajo, ya que será el lenguaje de programación central.

Muchos de estos entornos están relacionados con videojuegos producidos para la Atari 2600. Este entorno es un estándar en el campo del aprendizaje por refuerzo, donde se realizan numerosas pruebas y se comparan resultados entre diferentes algoritmos, como hemos visto en el caso de los desarrollados por David Silver.

En este trabajo se han probado numerosos de estos entornos, sobretodo en fases iniciales para familiarizarse con las características del aprendizaje por refuerzo. Más adelante, para la creación de nuestro propio entorno, se ha utilizado como base la librería de *Gym* por, como se ha comentado anteriormente, su fácil implementación

2.5. Conclusiones

Durante esta fase del estudio del estado del arte se han repasado los diferntes métodos dentro del aprendizaje por refuerzo y las diferentes inspiraciones para la realización de este trabajo. Como se ha comentado, una de las inspiraciones más importantes es David Silver con *Deepmind* y sus diferentes algoritmos. Estos algoritmos son prácticamente la base del aprendizaje por refuerzo y deben ser tenidos en cuenta para la realización de cualquier proyecto relacionado con el tema.

Por último se ha hablado acerca de *OpenAI* y su entorno para aprendizaje por refuerzo *Gym*. *OpenAI* se encuentra desarrollando numerosos proyectos de inteligencia artificial muy variados pero en los pasados años y actualmente han hecho mucho inca pié en el aprendizaje por refuerzo y la plataforma *Gym* ha podido permitir a muchos desarrolladores investigar en este campo y probar sus algoritmos, algo que es de agradecer a la compañía.

Todo esto ha servido de inspiración ya sea de forma directa o indirecta para el desarrollo positivo de este trabajo y durante la realización del mismo, debido a los constantes avances en el campo.

3

Diseño y Desarrollo

3.1. Introducción

Durante esta fase de diseño se realizarán todos los desarrollos de los módulos necesarios para llevar a cabo los experimentos que se realizarán en la siguiente fase. Entre estos módulos se encuentran el agente, el entorno, el sistema de recompensas y demás sistemas secundarios como la captación de los *embedings* de las fotos para su posterior procesamiento.

Cabe destacar que, aunque en esta fase es donde se desarrollarán estos sistemas, tendrán un constante desarrollo en fases posteriores, al analizar los resultados de cada uno de los experimentos y volver a reprogramar e incorporar cambios en las partes necesarias.

Previo al desarrollo y a modo de experimentos iniciales para realizar el diseño final, se estudiarán casos que corresponden a los ejemplos de la ya mencionada librería de *OpenAI*, que han sido de gran utilidad para entender el funcionamiento de los algoritmos de aprendizaje por refuerzo y a partir de estos, diseñar entornos y agentes propios.

3.2. Breve introducción sobre el aprendizaje por refuerzo

En esta sección, y de manera superficial, se realizará una introducción sobre los algoritmos de aprendizaje por refuerzo, de cara a aclarar términos que serán utilizados en secciones posteriores.

Dentro de los algoritmos de aprendizaje por refuerzo hay tres elementos fundamentales, el agente, el entorno y la recompensa. El agente se trata del elemento que aprenderá tras un número determinado de episodios o ejecuciones de un mismo programa. Este agente tiene que decidir en tomar determinadas acciones que, aplicadas al entorno que le rodea, le repercuten una recompensa positiva o negativa, en función del sistema de recompensas establecido, lo que provoca un proceso de aprendizaje.[3].

La parte positiva y novedosa, respecto a otros algoritmos de inteligencia artificial, es que durante este proceso de aprendizaje no se requiere de interacción humana, esta interacción se produce al inicio, al programar el entorno, el agente o el sistema de recompensas que va a seguir.

Este tipo de algoritmos ha sido usado en robótica pero también en juegos de mesa o, más recientemente, en videojuegos, como se ha comentado en el estado del arte. Esto es debido a que

en estos tipos de juegos tenemos un sistema de recompensas muy claro y cerrado, sin posibilidad de ambigüedad, lo que facilita el proceso de aprendizaje del agente.

Usando el ejemplo de los videojuegos, el entendimiento del aprendizaje por refuerzo es muy sencillo. El agente se trata del jugador principal, que en lugar de estar dirigido por un usuario, se dirige a si mismo. El entorno serían cada uno de los píxeles que vemos en pantalla en cada momento y el sistema de recompensas serían las propias mecánicas del juego. Al ser tan sencillo de aplicar, muchos de los ejemplos de aprendizaje por refuerzo son aplicados a videojuegos.

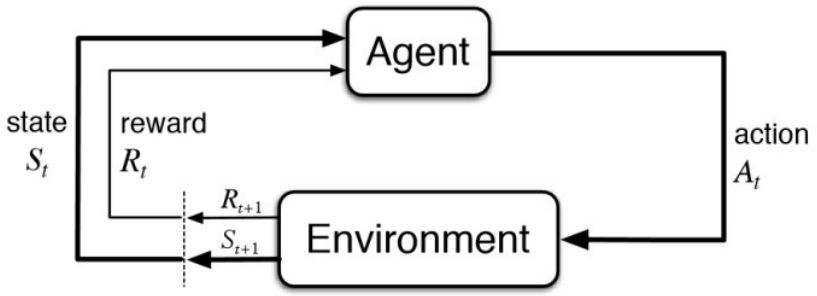


Figura 3.1: Esquema de los algoritmos de aprendizaje por refuerzo [3].

En la figura 3.1 podemos observar, de manera visual, el flujo de un algoritmo de aprendizaje por refuerzo. Como se ha comentado, el agente toma una acción que repercute directamente en su entorno y esto provoca una recompensa y un nuevo estado que se toman como entrada del agente para tomar la decisión de una nueva acción.

Cada ejecución de este bucle sería un episodio, una variable con importancia dentro de los experimentos realizados.

Para poder llevar un registro del rendimiento del agente durante las diferentes ejecuciones, la recompensa es un parámetro que se iba observando en tiempo real, según se producía, y además esta recompensa es almacenada en un vector, de tal forma que al final de todos los episodios, se realizaba una gráfica con la evolución de estas recompensas a medida que avanzaban el número de episodios ejecutados.

3.3. Experimento previo sobre ejemplo de *OpenAI*

Como se ha comentado anteriormente, previo al desarrollo propio de un agente o entorno, se ha procedido a realizar algunos experimentos sobre sistemas ya creados como los de *OpenAI*[9]. Esta empresa y su librería *Gym*, ya analizado anteriormente, posee una gran variedad de entornos con grandes atractivos visuales, que facilitan mucho el entendimiento del funcionamiento del aprendizaje por refuerzo.

Se basan en juegos sencillos, incluidos algunos famosos de videoconsolas retro, sobre los cuales, el usuario puede programar una serie de parámetros previos, como el número de episodios y observar los resultados posteriores. Algunos de ellos poseen un entorno gráfico, por lo que podremos ver, en tiempo real, el proceso de aprendizaje del agente, algo que resulta muy instructivo y que, a primera vista, resulta atractivo.

En el caso de este trabajo, se va a analizar un experimento en concreto, y es el llamado *LunarLander-v2* en la librería de *gym*. Este será el entorno que nos proporciona *OpenAI*, del cual no podremos modificar, a simple vista, nada, tampoco el sistema de recompensas, estará predefinido. También necesitaremos un agente, el cual nos proporciona *Gym*, o podremos crear

el nuestro propio e importarlo en el programa principal para que interactúe con el entorno de *Gym*.

En el caso de este trabajo, este experimento fue simple, se usó el entorno de *Gym* y el agente ya propuesto por ellos. Simplemente se debe ejecutar el programa principal (programa sencillo de 50 lineas de código) y observar en tiempo real los resultados de cada una de las "partidas"que realizará el agente. En este programa principal, como se ha comentado, el parámetro más sencillo de modificar y de entender es el número de episodios que jugará el agente. Un número pequeño de episodios supondrá menos oportunidades para que el agente sea capaz de entender el sistema de recompensas y el entorno que le rodea y un número demasiado grande supondría un gasto innecesario de tiempo y recursos, por lo que lo que nos plantea *Gym* es encontrar el equilibrio perfecto y alcanzar los mejores resultados posibles.

El experimento consiste en un módulo lunar cuyo objetivo es el aterrizaje entre las dos banderas que hay en la superficie. Dependiendo del lugar del aterrizaje recibirá una mayor o menor recompensa, incluso una recompensa negativa a modo de penalización.

El usuario que ejecuta este experimento no tiene acceso (en primera instancia) al sistema de recompensas, y esto es algo que ha motivado también el desarrollo de este trabajo, el crear un entorno desde cero y poder crear y modificar el sistema de recompensar.

Al realizar este experimento, se puede ver como el módulo lunar, al principio, ejecutará movimientos aleatorios, al no entender el sistema de recompensas. Incluso hay momentos en los que el agente, al ver que obtiene recompensas negativas al aterrizar, se quedará en el aire un tiempo. Pero a medida que se realizan episodios (cada vez que se obtiene una recompensa, el juego se reinicia y el módulo lunar vuelve a la posición inicial) los resultados van siendo cada vez mejores.

Por último, y como modificación a este programa de *Gym*, se ha eliminado la parte de renderización (para no consumir tantos recursos) y se han guardado los datos de las recompensas a lo largo de los episodios en una gráfica para observar su evolución. En la figura 3.2 se pueden observar los resultados obtenidos.

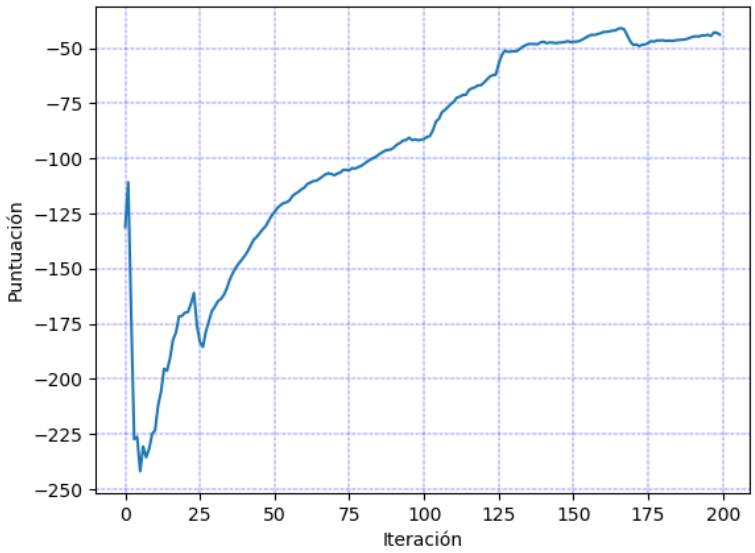


Figura 3.2: Gráfica resultante del experimento con la librería Gym.

3.4. Desarrollo del programa principal y del entorno

Como se ha comentado en anteriores secciones, el algoritmo de aprendizaje por refuerzo se basa en dos elementos fundamentales, el agente y el entorno. Durante esta sección se explicarán las características del entorno creado y del programa principal que ejecutará todo el flujo de aprendizaje del agente.

Dentro del programa principal inicializaremos el agente, por lo que desde ahí podremos modificar su parámetros, así como también modificar el número de episodios, como se ha comentado en el experimento previo con el ejemplo de *OpenAI*.

De la misma forma, en el programa principal, inicializaremos la variable del entorno, a modo de objeto de una clase creada con sus propias funciones que se comentarán más adelante.

Por lo tanto el flujo de ejecución del programa principal es el siguiente:

- Se establecen los parámetros del agente y se inicializa (se profundizará en sus parámetros en la descripción del agente)
- Se inicializan los vectores que almacenarán las recompensas, así como se establece el número de episodios
- Comienza el bucle principal que se ejecuta según el número de episodios.
- Llamada a la primera función del entorno *reset()*: crea un objeto de la clase *PyGame2D* (eliminando previamente el que había antes) y de este objeto llama a la función *observe*. Esto lo que hace simplemente es, al crear el objeto, extrae un vector de *embedings* de una de las fotos (extraída al azar) de la base de datos y mediante la función *observe* lo devuelve, de tal forma que en el programa principal ya tenemos cargado el vector de *embedings* de las imágenes.
- Después se comprueba que el estado de la partida no esté finalizado, en la mayoría de los experimentos realizados para cada foto el agente tomaba una decisión y se calculaba la recompensa y entonces el estado de la partida finalizaba, antes de tomar la siguiente foto.
- A partir del modelo facial, el cual da como resultado un vector de *embedings*, el agente tomará una decisión de la acción que va a realizar. La descripción de este proceso se realizará más adelante.
- Dada esta acción, se llamará a la siguiente función del entorno, la función *step*. Esta es la función más importante, ya que se encargará de, dada la acción realizada por el agente, devolver la recompensa que obtiene, además de indicar que el estado de la partida ha finalizado y de cual será el siguiente vector de *embedings* sobre el que tendrá que realizar una acción el agente en el episodio siguiente. El cálculo de la recompensa varía según el experimento que se realiza, pero se basa en tres factores. La acción que realiza el agente, la acción que realiza la máquina, es decir, el contrincante del agente (esta decisión puede ser aleatoria, o seguir un patrón definido para realizar un experimento en concreto) y las reglas que se han programado para la obtención de esta recompensa.
- Una vez ya se ha obtenido la recompensa, lo que queda es el proceso de aprendizaje del agente, que se describirá en la sección correspondiente y el almacenamiento de la recompensa (o puntuación) de este episodio en su vector para que, al final de todos los episodios, sea representado.
- La parte de la representación es relativamente sencilla, haciendo uso de las funciones de la librería de *matplotlib.pyplot* modificando leyendas, ejes, umbrales, etc ...

Todo este flujo de ejecución se puede ver reflejado, de manera más visual, en la figura 3.3:

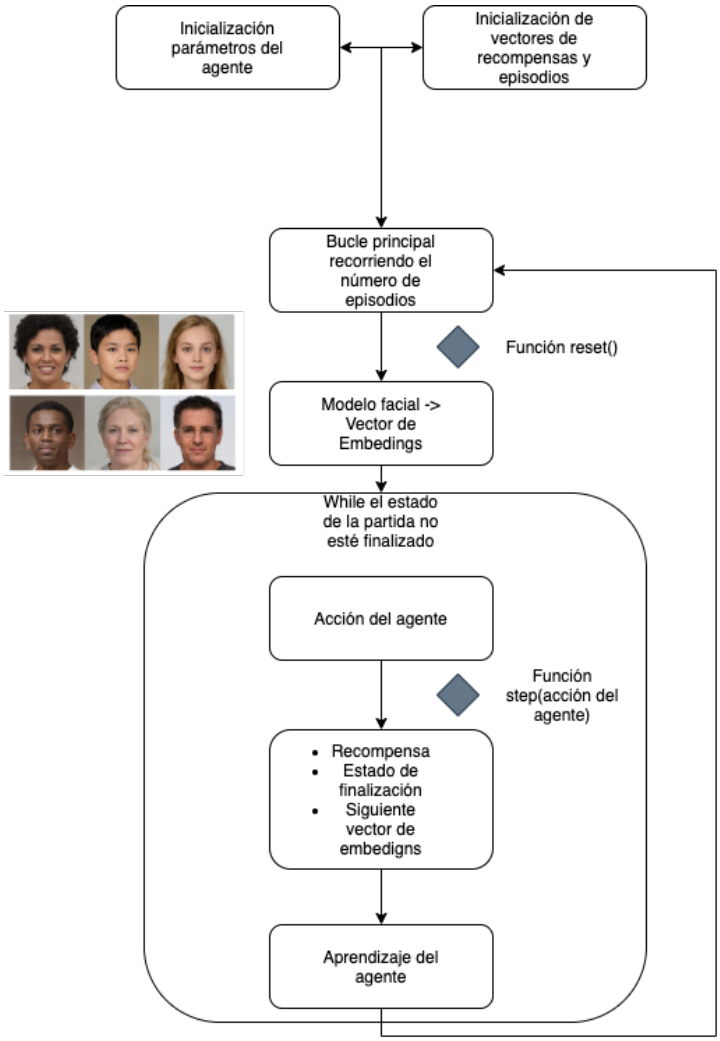


Figura 3.3: Esquema del flujo de ejecución del programa principal.

3.5. Desarrollo del agente

Durante esta sección, al igual que se ha realizado con el entorno y con el sistema de recompensas, se analizará el desarrollo del agente.[10] De la misma forma que antes, primero se enunciarán los diferentes elementos y después se representará de manera más visual, a medida que avanza el flujo del programa principal.

Antes de comenzar con la descripción del agente, es necesario comentar la técnica utilizada, dentro del aprendizaje por refuerzo para el aprendizaje automático del agente. Esta técnica se trata del *Q-Learning*[9]

Este algoritmo es de suma importancia dentro del proceso de aprendizaje, ya que se encarga de establecer unas normas que permitan que el agente maximice su recompensa eligiendo la acción más correcta, es decir, este algoritmo es el propio proceso de aprendizaje.

En la figura 3.4 podemos observar su fórmula que se encuentra directamente aplicada en el agente (posteriormente se comentará en que parte se encuentra y cuando es ejecutada). A continuación se explicarán los parámetros que componen esta fórmula.

$$Q^{new}(s_t, a_t) \leftarrow (1-\alpha) \cdot \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \overbrace{\left(\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}} \right)}^{\text{learned value}}$$

Figura 3.4: Fórmula del algoritmo de *Q-Learning*

Existen dos variables en la fórmula que deben inicializarse con un valor y son las siguientes:

- El factor de descuento, γ: Permite regular el impacto de la recompensa futura en el agente, si es más cercano a 0 el agente solo "mirará"por una recompensa inmediata, si es más cercano a 1, tendrá más importancia la recompensa futura.
- El ratio de aprendizaje, α: Este índice nos permite balancear entre el aprendizaje o no, es decir, un factor de 0 (como se puede observar en la fórmula) haría que el valor de Q tomara el antiguo valor, y un valor de 1 haría que el valor de Q se centrara sólo en los nuevos estado y valores futuros. Hay que considerar que nuestro entorno es totalmente determinista, ya que el azar no debería formar parte de las decisiones del agente, por lo que es necesario que este ratio sea 1, es más, en el código la fórmula solamente se compone de la segunda parte.

Entre el resto de parámetros de la fórmula se encuentran: la recompensa obtenida y el parámetro Q, que no es más que el resultado de esta fórmula en resultados anteriores, ya que se trata de una fórmula iterativa, en el que este valor de Q se va actualizando en cada estado. Una vez se han comentado los parámetros fundamentales de la fórmula, el funcionamiento del algoritmo es el siguiente: Primero el agente comienza realizando una acción aleatoria, inicializando el valor de Q y estableciendo una recompensa inicial y un estado siguiente, en base a esta decisión aleatoria. El siguiente paso es simplemente una actualización en bucle de este valor de Q, en base a la recompensa obtenida, el siguiente estado (en una estimación del valor máximo en base a una tabla de acciones y el factor de descuento que se ha comentado anteriormente).

Una vez se ha comentado el funcionamiento del algoritmo de *Q-learning*, su aplicación en el agente de este trabajo es una modificación del mismo, añadiendo una parte que incorpora redes neuronales, lo que da lugar al nombre de *Deep Q-learning*.[4]

El funcionamiento del algoritmo es el mismo, pero se incorpora una red neuronal para simplificar el trabajo del algoritmo y ahorrar recursos en su ejecución. Se ha comentado que dentro de la fórmula, una vez tenemos el valor del siguiente estado, se extrae de una tabla el valor de Q más óptimo, para cada una de las acciones del agente. Este es un proceso muy laborioso si existen multitud de estados o acciones o para una gran cantidad de ejecuciones con vectores de entrada diferentes. Es por lo que este valor de Q máximo, será obtenido mediante una red *dqn* a partir del valor del siguiente estado.

De esta forma, el programa no necesita buscar en esta tabla o el actualizarla en cada ejecución, simplemente toma la salida de la red como valor para, aplicándolo a la fórmula, actualizar el valor de Q y continuar con el proceso de aprendizaje.

En la figura 3.5 podemos observar una comparación gráfica entre la aplicación del algoritmo de *Q-Learning* y su variante mediante redes neuronales.

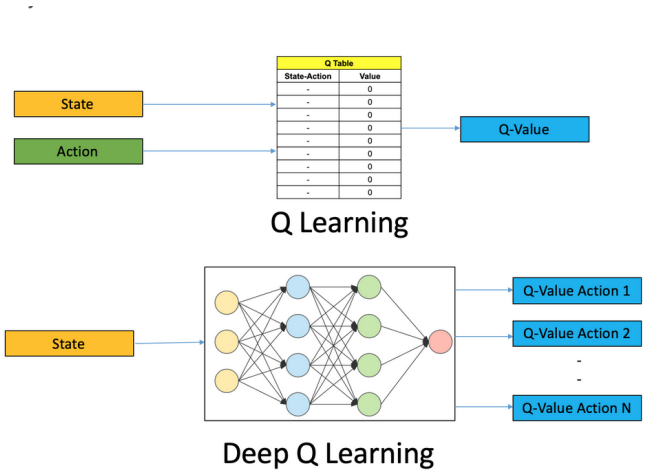


Figura 3.5: Comparación entre *Q-Learning* y *Deep Q-Learning*[4].

Una vez se ha estudiado el algoritmo o conjuntos de algoritmos que utiliza el agente en el proceso de aprendizaje, será más sencillo comentar la descripción en sí del mismo, como es su ejecución dentro del flujo del programa principal y cuáles son sus variables iniciales.

Inicialización:

- Se inicializan los parámetros del agente, entre los que se encuentran el factor de descuento, el número de acciones que podrá realizar el agente, el nombre de la red neuronal (para guardar su ejecución) y un parámetro denominado *batch size*, fijado a 64 y que indica que el agente aprenderá pasadas 64 ejecuciones, hasta entonces guardará las recompensas y estados.

- De la misma forma creamos la red neuronal convolucional con dos capas densas y que depende del número de acciones que pueda tomar el agente

Ejecución del programa

- Una vez tenemos el vector de *embedings*, el agente deberá tomar la decisión de la acción que realizará. Esto se realiza mediante la función *chose action* que depende de este vector, y predice la acción siguiente mediante la red neuronal, utilizando como input el vector. Se debe recordar que en la primera ejecución esta acción es aleatoria (existe una variable para distinguir este caso).

- Una vez se tiene la recompensa y el estado siguiente en base a la acción que ha tomado el agente, se almacena esta información mediante la función *remember*. Simplemente se guardan en unos *arrays* que se han inicializado al iniciar el agente y que sirven para almacenar todos estos valores, ya que hasta que se alcanza la ejecución 64 el agente no realiza ningún proceso de aprendizaje, si no que lo realiza con la información de las pasadas 64 ejecuciones.

- Finalmente y tras 64 ejecuciones del programa comienza el periodo de aprendizaje. Se recoge la información que se ha almacenado y se tendrían los siguientes elementos. Un vector de estados actuales y sus correspondientes estados futuros, de tamaño 64 x número de atributos que hay en cada estado según el entorno, un vector de recompensas de tamaño 64, un vector de acciones de tamaño 64 x número de acciones que puede realizar el agente y un vector de finalización de tamaño 64 que recoge el estado de la partida (valor binario de si ha finalizado o no). Con toda esta información de estados, acciones y recompensas podemos realizar el algoritmo de *Q-Learning* y realizar el proceso de aprendizaje cada 64 episodios.

En la figura 3.6, se puede observar, de una manera más visual, la descripción del desarrollo del agente.

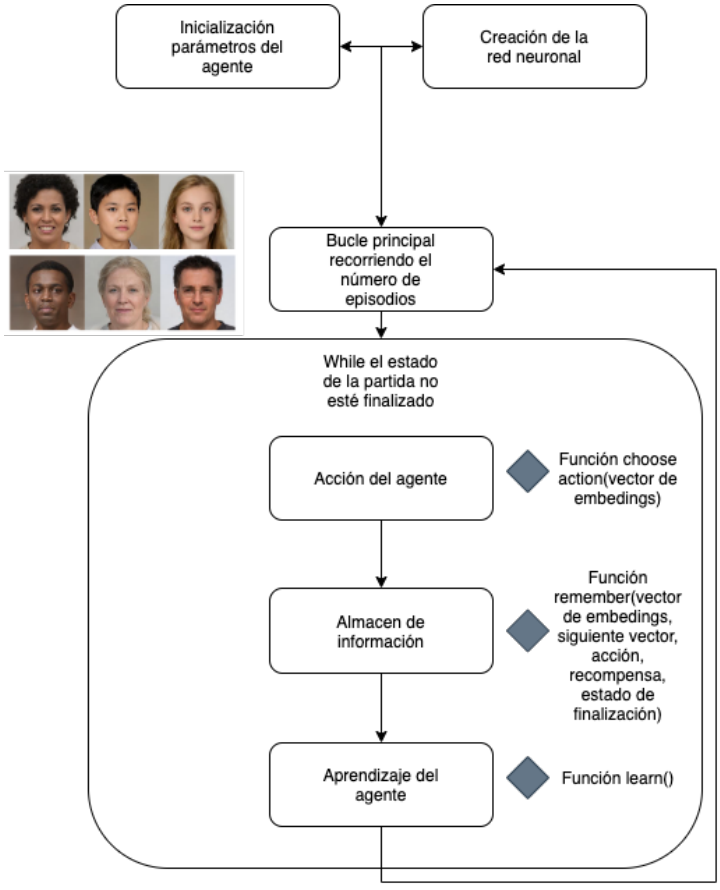


Figura 3.6: Esquema del flujo de ejecución del agente.

3.6. Modelo Facial para la caracterización de la información a partir de imágenes de la cara

En esta sección se comentará acerca de uno de los sistemas más importantes de la realización de este trabajo.

A pesar de que se han observado modificaciones específicas para este trabajo, la estructura de un agente y un entorno con su sistema de recompensas es algo común en todos los sistemas de aprendizaje por refuerzo. En cambio, el hecho de incorporar un vector de *embedings* como input para el agente, es decir, como entorno propiamente dicho, es algo novedoso y le añade complejidad al algoritmo de aprendizaje por refuerzo.

Durante esta sección se comentará acerca de las imágenes usadas así como el sistema que obtiene sus vectores, algo que será común para todos los experimentos realizados.

Por motivos de privacidad y por la gran cantidad de imágenes necesarias, se ha optado por la utilización de imágenes sintéticas. En concreto, el sitio web *Generated Photos*[11] cuenta con una serie de bases de datos ya creadas mediante inteligencia artificial de imágenes sintéticas. Estas imágenes han sido creadas a partir de fotos que existen realmente, pero gracias a un proceso de redes neuronales generativas adversas, se obtienen estas imágenes completamente artificiales. La ventaja es que el usarlas no incumple ninguna normativa como la GDPR[12] y tienen una excelente calidad. Se incluyen 10000 imágenes (base de datos gratuita para fines académicos) de 128x128 píxeles con sus respectivos metadatos, que proporcionan información sobre la etnia, el género, la edad, color de ojos, de pelo, etc...

Figura 3.7: Visión general de las imágenes sintéticas.

Dentro de la base de datos, existe una gran variedad de imágenes con rasgos completamente diferentes, algo que será utilizado para la realización de distintos tipos de experimentos.

Figura 3.8: Variación de rasgos en imágenes de género femenino.

Figura 3.9: Variación de rasgos en imágenes de género masculino.

En las figuras 3.8 y 3.9, se pueden observar más detalladamente ejemplos de estas imágenes sintéticas. Se puede observar como, para una imagen con unos rasgos determinados, se pueden variar parámetros como el color del pelo, de la piel o las emociones. Todo esto de forma sencilla desde la herramienta de *Generated Photos*. Por ejemplo, en el caso de la figura 3.9, se puede observar a la izquierda un hombre de edad media, piel blanca y pelo oscuro. Las imágenes situadas a la izquierda son variaciones de esta primera foto, en el pelo, tono de piel... Como se puede apreciar, se obtienen resultados muy realistas y que, en todos los casos, es prácticamente imposible afirmar que se tratan de imágenes sintéticas. Ninguna de las imágenes que se han observado o que componen la base de datos son de personas reales y, aun así, el realismo es tan alto, que los resultados obtenidos pueden ser comparados con cualquiera en el que usen imágenes de personas reales

Ya se ha observado en que parte del bucle de ejecución se obtiene el vector de *embedings*, esta es la parte donde accedemos a las imágenes de la base de datos. Generalmente en los experimentos, se escoge una de las imágenes al azar, y para el nombre de esa imagen en cuestión, se busca en un diccionario el vector de *embedings* de los meta datos de esa imagen. Es un proceso sencillo, este vector es la información que el agente "ve"para tomar la decisión de que acción ejecutar en cada episodio.

Por último, destacar que en esta misma sección de código, donde se extrae el vector de *embedings*, es donde la "máquina", es decir, el contrincante del agente, decidirá que acción realizará. El término "decidirá"no es del todo correcto, ya que la acción de la "máquina"se encuentra totalmente controlada por el programador, al contrario que el agente, y varía según el experimento realizado.

Para la mayoría de experimentos realizados, la acción de la máquina va a seguir una distribución de *Bernouilli*[13] (se explicará en secciones siguientes), en la que el programador puede establecer el porcentaje de episodios en el que la máquina tomará una acción, aunque no controlará en que episodios en concreto va a realizar dicha acción. Esto se ha usado para asemejarlo lo máximo posible a la realidad, en el que las personas pueden llegar a tener ciertos comportamientos dispares y no siempre realizar las mismas acciones y en el mismo orden.

Explicada esta sección se podrán entender de una manera más completa los experimentos

posteriores realizados.

3.7. Conclusiones

Como se ha podido comprobar, este capítulo ha sido de gran importancia en la realización de este trabajo. En este capítulo se incluyen todos los desarrollos que harán posible la realización de los experimentos durante el resto de trabajo.

Hay que destacar algo importante, que se ha comentado al inicio, y es que, aunque en este capítulo se ha descrito el desarrollo de todos los sistemas, estarán en constante desarrollo a medida que se realizan las diferentes pruebas, ya que será necesario modificar valores del agente, del entorno, o del sistema de recompensas.

Este capítulo también es importante ya que se ha realizado una breve introducción sobre el aprendizaje por refuerzo, para introducir conceptos clave y describir sus elementos fundamentales, algo que ahorrará explicaciones en secciones posteriores.

Se ha realizado una descripción de un primer experimento realizado que se ha decidido no incorporar en el capítulo siguiente al tratarse de una demostración y acercamiento al aprendizaje por refuerzo a través de una librería ya desarrollada.

Por último se han descrito los tres elementos fundamentales de este trabajo, el agente, el entorno y el sistema de captación de los datos de las imágenes. Todo ha sido desarrollado para este trabajo y se ha descrito en forma de diagramas para facilitar el entendimiento de todos los módulos de programación.

En el siguiente capítulo, como ya se ha comentado, se expondrán los experimentos realizados y se analizarán los resultados obtenidos.

4

Experimentos Realizados y Resultados

4.1. Introducción

Esta sección es la parte central de este trabajo, ya que recoge todos los experimentos y pruebas que se han realizado, variando los parámetros y sistemas desarrollados, en busca del mejor resultado posible y de situaciones concretas que resulten interesantes para el desarrollo del trabajo.

Se parte del anterior capítulo, en el que se han desarrollo todos los sistemas necesarios para ejecutar pruebas mediante el algoritmo de aprendizaje por refuerzo.

Durante el desarrollo de este capítulo se irán describiendo cada una de las pruebas realizadas, exponiendo los parámetros que definen esas pruebas y finalmente observando los resultados obtenidos.

Las pruebas han seguido un orden en concreto, en base al desarrollo del trabajo. Al comienzo se realizan pruebas más simples, con un entorno que no cuenta con imágenes ni vector de *embedings*. Después se irán incorporando estas imágenes pero con sistemas de recompensas sencillos, para terminar con un experimento que incorpore todo lo aprendido y que, además, tenga un sistema de recompensas basadas en *el dilema del prisionero*[14], el cual se explicará más adelante.

4.2. Primer experimento: Agente diferencia entre 0 y 1

Como sugiere el título, el objetivo principal de este experimento es que el agente desarrollado, pueda aprender a diferenciar entre 0 y 1. Para ello, el sistema de recompensas es simple, si el agente elige el mismo número que la máquina (su contrincante, que no tiene ningún proceso de aprendizaje) se le recompensará con un punto, si elige uno distinto no sumará puntos. El agente podrá diferenciar entre 0 y 1 ya que estas serán las dos únicas elecciones posibles del agente y de la máquina, se trata de un sistema de decisión binario.

Para que se pueda observar el proceso de aprendizaje del agente, la máquina elegirá 1 el 100 % de las veces. Lo que se podrá apreciar es que el agente, en un inicio, realizará acciones completamente aleatorias, lo que le llevará a recibir recompensas variables. Todo esto hasta llegar

a un cierto número de episodios en el cual "se dé cuenta"de que elegir 1 siempre le proporciona la máxima recompensa posible. Y es que este es el objetivo fundamental del agente, encontrar la acción que, realizando el mínimo proceso de decisión posible, se obtenga la máxima recompensa.

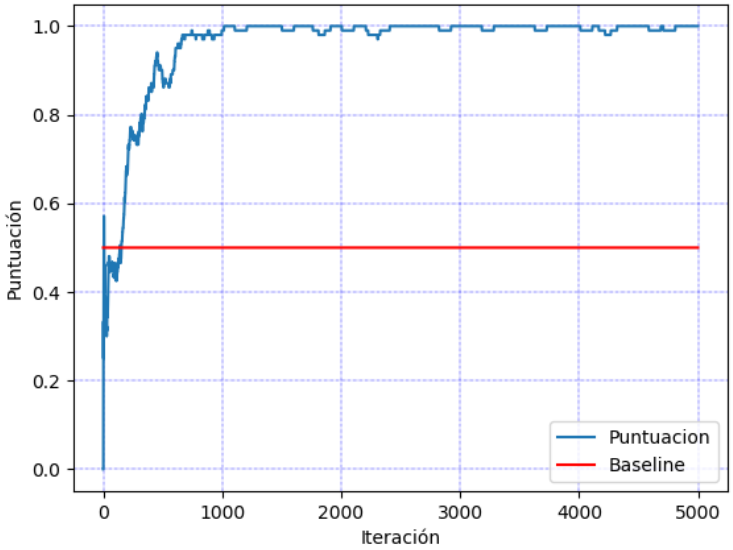


Figura 4.1: Resultados del primer experimento.

En la figura 4.1 se pueden observar los resultados obtenidos. En el eje de ordenadas se encuentra la puntuación media acumulada en los diferentes episodios, siendo 1 la máxima posible (al tratarse de una media) y un 0 la mínima. En el eje de abscisas se encuentran las iteraciones, es decir, los episodios que el agente lleva a cabo, numerados.

En cada episodio el agente toma una decisión, se calcula una recompensa en base a esa decisión, se guardan los resultados de esa ejecución y se pasa al siguiente episodio.

La linea roja denominada *baseline*, es la linea que marca la aleatoriedad. Al tratarse de un sistema binario, si el agente realizará acciones aleatorias continuamente, obtendría una recompensa un 50 % de las veces. Para considerar un resultado como positivo, el objetivo es situar las puntuaciones por encima de este umbral.

Observando la gráfica se puede observar que el agente alcanza esta puntuación media de 1, es decir, se podría decir que aprende lo que es el 1 y el 0, ya que "sabe"que si la máquina ha elegido el 1 (es lo que tiene el agente como input) el agente debe elegir un 1 también y no un número aleatorio. A pesar de esto se puede observar como el proceso es lento, el mejor resultado posible no se alcanza hasta las 1000 ejecuciones, y esto, contando con que es una decisión simple (dos valores posibles) y un sistema de recompensas sencillo.

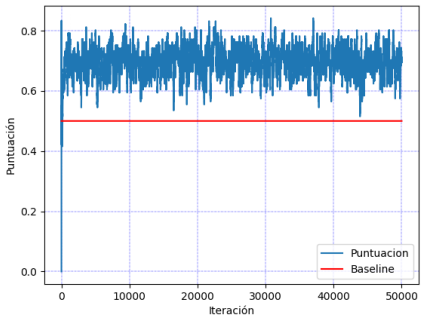
La lentitud de alcanzar los mejores resultados posibles será una constante que se observará en la mayoría de los experimentos posteriores y puede llegar a ser un punto negativo si se compara con otros algoritmos de aprendizaje supervisado. Pero su punto positivo, algo propio del aprendizaje por refuerzo y que se ha comentado anteriormente, es que requiere prácticamente nula interacción humana.

4.3. Segundo experimento: Agente sin entorno

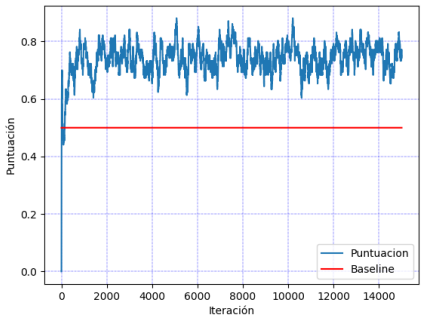
Este experimento va a seguir los pasos fundamentales del anterior. No se ha incorporado aún el entorno con el vector de *embedings* de las imágenes. En la anterior prueba, estaba programada para que el contrincante del agente eligiera 1 el 100 % de las veces. El agente sólo podía tomar dos decisiones posibles (0 y 1) y este factor continua en este experimento. En este caso el çontrincante"no elegirá 1 siempre, si no que, aunque predominará una decisión, tendrá una determinada variación. Esto hace que se asemeje a la realidad y a un contrincante humano, el cual, no tomará siempre las mismas decisiones. Esto también hará, como se comprobará en las gráficas, que la puntuación no alcance los buenos resultados que se alcanzaron en el anterior experimentos

Para definir la elección del contrincante, se usará una distribución de *Bernouilli*[13]. Gracias a esta distribución, como programador se podrá definir el porcentaje (la probabilidad) de que se tome la decisión de elegir 1 y por lo tanto, también de elegir lo contrario.

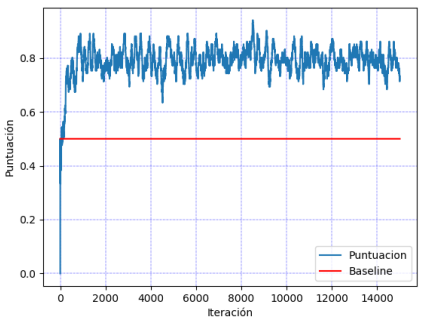
Se van a realizar 3 variaciones en este experimento, variando este porcentaje de elección un 70 %, 75 % y 80 %. Es decir, para el primer caso, el contrincante del agente elegirá 1 el 70 % de las partidas.



(a) Resultados probabilidad Bernoulli 70 %



(b) Resultados probabilidad Bernoulli 75 %



(c) Resultados probabilidad Bernoulli 80 %

Como se puede observar, de forma común, en los tres resultados, se supera en todos el umbral definido, que se ha explicado en el anterior experimento. Esto es algo positivo, ya que en ningún caso el agente tiene un comportamiento aleatorio, es más, se puede apreciar una puntuación media acorde con la probabilidad de *Bernoulli* establecida.

Esto es algo que tiene sentido, ya que, como se ha comentado anteriormente, el agente maximiza la mayor recompensa posible con la menor complejidad posible de decisión. Esto provoca que pasadas unas 1000 iteraciones, el agente sea capaz de ver que elegir 1 siempre

maximiza sus resultados, lo que, por ejemplo en el caso de la figura 4.2a, hace que en el 70 % el agente obtenga la recompensa positiva y en el resto de casos no, provocando así una puntuación media de al rededor de 0.7 sobre 1, un 70 %. Lo mismo ocurre para el resto de los caos.

4.4. Tercer experimento: Reconocimiento de género

Este experimento es diferente a los realizados anteriormente. Ahora si que existe un entorno como tal, en el cual se basará el agente para tomar sus decisiones. Antes sólo tenía la información de la recompensa obtenida, lo que le incentivaba a tomar una decisión u otra y mejorar con el paso de las iteraciones. Ahora tendrá una mayor cantidad de información, las imágenes, es decir, sus vectores de *embedings*, además de la recompensa.

Para el proceso de obtención del vector de *embedings*, se han seguido los pasos desarrollados en la sección 3.6. El proceso es el descrito a continuación. Primero, mediante un bucle que recorre cada una de las iteraciones establecidas, se extraerá una imagen al azar de la base de datos. No se sigue ningún tipo de preferencia, simplemente se extrae la imagen. Para cada imagen de la base de datos existe un archivo *json* con los metadatos correspondientes a esa imagen. Entre estos metadatos se encuentran parámetros como a qué género pertenece la persona de la imagen, color de pelo, ojos, emociones que tiene la persona en el momento de la imagen, etnia a la que pertenece, etc... Todos estos parámetros expresados en porcentajes. Es decir, para una imagen en la que salga una persona de género masculino, tendrá un porcentaje de hombre asociado mayor que el de mujer, de esta forma, se distingue por los metadatos.

Una vez se tiene acceso al parámetro deseado de una de las imágenes, en función de este parámetro, el contrincante del agente ejecutará una acción u otra (elegirá 1 o 0 en este caso). Todo esto permite crear patrones en función de rasgos observados en las caras de las personas. Por ejemplo, este sistema se puede usar para que, forzado artificialmente, los hombres elijan siempre 1 y las mujeres 0, de tal forma que el agente pueda encontrar este patrón y aprovecharlo para obtener la mayor recompensa posible.

Se establecerán patrones con mayor complejidad que el ejemplo anterior, todo para acercar el sistema a la realidad, en el que pueden existir patrones de comportamiento en personas con determinados rasgos. Estos patrones pueden llegar a ser una clara ventaja para los algoritmos de aprendizaje por refuerzo, ya que si el agente es capaz de encontrar estos patrones a lo largo de las iteraciones, maximizará los resultados obtenidos en iteraciones futuras.

Pero, primero, se ha querido ejecutar una prueba en la cual, para cada imagen de la base de datos, el contrincante del agente realice una decisión aleatoria, para saber como se comporta el agente y observar los resultados de uno de los peores escenarios, desde el punto de vista de la recompensa esperada.

En la imagen 4.3 se puede observar el resultado de esta prueba. Como se puede observar, la media de resultados es de 0.5 sobre 1, es decir, un 50 % de las veces el agente conseguirá una recompensa positiva y el otro 50 % restante. Esto es un comportamiento totalmente aleatorio, para lo cual no es necesario ningún tipo de aprendizaje.

Si estos fueran los resultados reales, se podría considerar un experimento fallido. Pero a pesar de que el contrincante del agente (se va a denominar "la máquina.ª partir de ahora, por simplificación) ha seguido un comportamiento aleatorio, esto no es lo que ocurre en la realidad. En la realidad, en un "juego"de decisión binario, una persona no seguiría comportamientos aleatorios del todo, si no que existen patrones, en base a multitud de factores relacionados con la persona, que le llevan a modificar su comportamiento. Este es el objetivo fundamental de este trabajo, encontrar, en esos patrones de conducta, una oportunidad para el aprendizaje por refuerzo, de obtener la máxima recompensa posible.

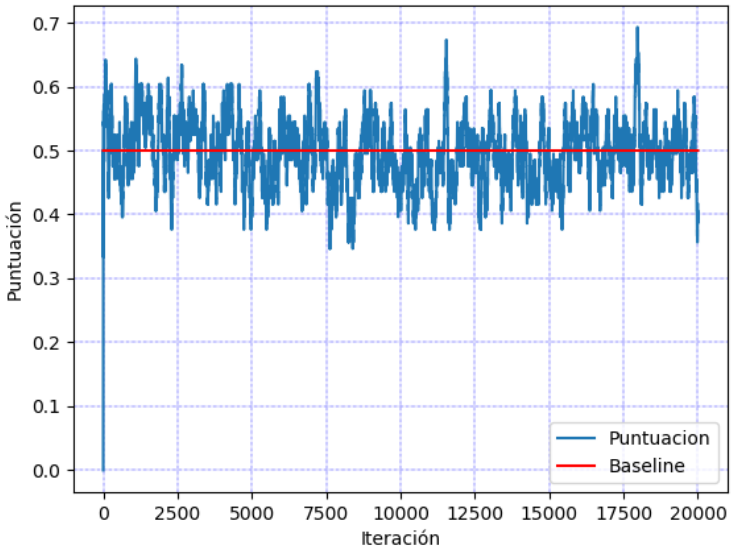


Figura 4.3: Resultados para una elección aleatoria del contrincante del Agente.

Es por esto por lo que se ha establecido un patrón artificial, uno en concreto para el caso de este experimento. Como se encuentra definido en el título de esta sección, el patrón se encuentra en torno al género de la persona de cada imagen. Y es que al extraer una de las imágenes al azar de la base e datos, se pueden acceder a sus metadatos. En ellos, en un formato *json* se encuentra multitud de información relacionada con la imagen, como es el caso del género

En cada imagen, como se ha comentado, se encuentra un porcentaje que define cuanta probabilidad hay de que la persona sea de género masculino o femenino. Por lo tanto para determinar la decisión de la máquina, si esa probabilidad es mayor en el género masculino que femenino, decidirá elegir el 1, y en el caso contrario, un 0. De esta forma no se trata de un comportamiento aleatorio, todo lo contrario, todos los hombres elegirán 1 y las mujeres 0. Ahora se va a observar cuantos episodios son necesarios para que el agente encuentre este patrón o si es siquiera capaz de hacerlo y alcanzar la máxima recompensa posible.

Hay que recordar que el sistema de recompensas se mantiene, si el agente elije el mismo número que la máquina, obtendrá 1 punto de recompensa, en caso contrario, 0 puntos.

Los resultados obtenidos se encuentran en la figura 4.4. Como se puede observar, se alcanzan unos resultados muy positivos en pocas iteraciones, aproximadamente 1000, pero durante los 20000 se observa una ligera oscilación de los resultados. A pesar de esto, se puede apreciar una tendencia ascendente de los valores de la recompensa obtenida con una menor oscilación a medida que aumenta el número de episodios. Estos resultados obtenidos son positivos y significa que el agente es capaz de reconocer el género, es decir, encontrar este patrón de decisiones en base a este rasgo y aprovecharlo en futuras iteraciones para obtener la recompensa positiva. La puntuación oscila entre 0.9 y 1, lo que significa que que es prácticamente una recompensa positiva, por que, hay que recordar, que esta recompensa es la recompensa media, que tiene en cuenta las primeras iteraciones, en las que hay resultados peores, debido a que el agente no ha identificado el patrón de este experimento.

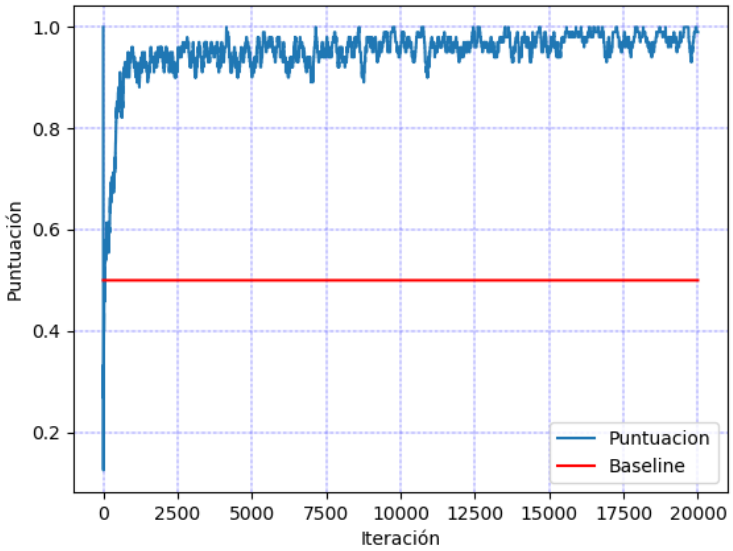


Figura 4.4: Resultados para una decisión en función del género.

Para completar este experimento, se va a modificar este patrón de decisión en torno al género. Se creará un patrón más realista. En el caso del género masculino, no se tomará la decisión de 1 para todas las personas con este género, si no que lo harán un determinado porcentaje de estas. Esto lo acerca más a la realidad ya que, puede llegar a existir un patrón en torno al género, pero no puede cumplirse siempre al 100 %.

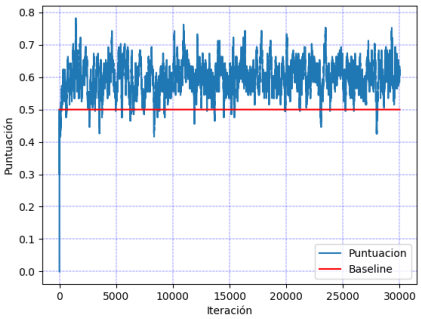
Para establecer este porcentaje se hará uso, al igual que en el experimento anterior, de *Bernoulli*. Ya se ha comentado acerca de este proceso así que a continuación se encuentran las gráficas de los resultados para una probabilidad de un 70 %, 75 % y 80 % de que, para el género masculino se escoja un 1. En el caso femenino sería lo mismo pero la probabilidad contraria (para elegir el 0).

Como se puede apreciar a simple vista, los resultados son inferiores al anterior caso, algo que es de esperar. A pesar de esto, en todos los casos se obtiene una media de puntuación superior a la aleatoriedad, algo que resulta positivo. También se pueden observar como los resultados mejoran a medida que la probabilidad de *Bernoulli* se acerca al 100 %. Pasando de una media de 0.6 en la primera gráfica a una media de 0.7.

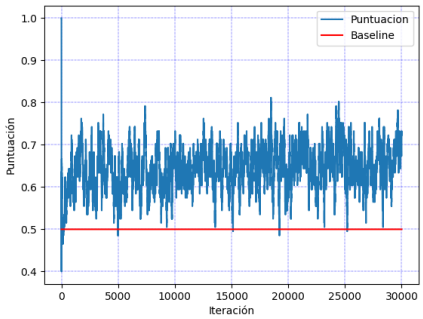
Por último, como prueba final en torno al parámetro del género, se ha estudiado el impacto que supone cambiar la arquitectura de la red neuronal.

Hasta ahora, en todas las ejecuciones, la estructura de la red neuronal no ha cambiado. Estaba formada por una primera capa densa que tiene la dimensión del vector de *embedings* que tiene como input, en el caso de las imágenes de la base de datos, de tamaño 512. A continuación se encuentra una activación de tipo rectificador ($ReLU$)[15], una función de activación muy común y usada por su eficiencia, ya que permite que, para valores de salida menores a 0, no se active la neurona, permitiendo, así, que un número limitado de ellas estén activas.

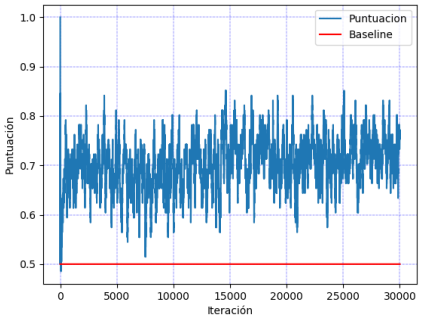
A continuación se encuentra la última de las capas densas, dependiente del número de acciones como tamaño, es decir 2 tipos de acciones en este caso. Y por último una activación final, de tipo *Softmax*. Este tipo de activación final es también muy común, sobre todo en redes neuronales con imágenes, ya que trasforma la salida de la red en una distribución de probabilidad, por lo tanto sus valores quedan entre 0 y 1, mucho más simple de clasificar.



(a) Resultados para una decisión en función del género con probabilidad Bernoulli 70 %



(b) Resultados para una decisión en función del género con probabilidad Bernoulli 75 %

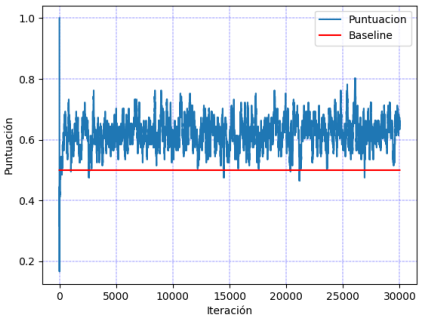


(c) Resultados para una decisión en función del género con probabilidad Bernoulli 80 %

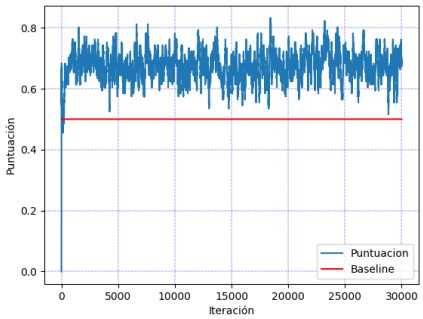
Los resultados correspondientes a esta estructura se pueden observar en las figuras 4.5a, 4.5b y 4.5c. Se va a realizar una prueba similar, pero variando esta estructura, sustituyendo esta última activación por otra capa densa de tamaño 256, que simplifique la red, seguida de una activación *ReLU*, al igual que la primera capa densa, para terminar con la última capa densa del tamaño del número de acciones, 2.

Se pretende estudiar si, el hecho de incluir una capa densa más, algo computacionalmente más costoso, puede beneficiar a los resultados obtenidos.

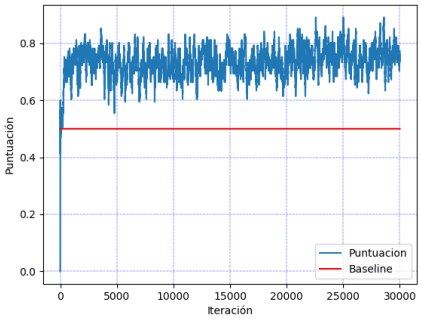
Con un estudio superficial, se puede afirmar que para una probabilidad de *Bernoulli* de un 70 % no se puede apreciar diferencia. Sin embargo, según aumenta esta probabilidad si que se pueden observar mejores resultados de forma ligera, especialmente para el último caso, en el que se puede comprobar un comportamiento en cuanto a la puntuación obtenida, claramente ascendente y superior al caso que se había estudiado anteriormente. A pesar de esto, los cambios no son demasiado significativos y, en cualquier caso, los resultados se encuentran por encima del umbral de la aleatoriedad.



(a) Resultados para una decisión en función del género con probabilidad Bernoulli 70 % y dos capas densas



(b) Resultados para una decisión en función del género con probabilidad Bernoulli 75 % y dos capas densas



(c) Resultados para una decisión en función del género con probabilidad Bernoulli 80 % y dos capas densas

4.5. Cuarto experimento: Reconocimiento de la etnia

Este experimento realizado tiene características similares al anterior que se ha realizado. Se trata de un experimento complementario para observar si, con un rasgo más complejo de detectar como es la etnia, se obtienen, igualmente, buenos resultados. Durante este experimento no se van a hacer variaciones en la red neuronal ya que es algo que se ha probado en el anterior experimento. Simplemente se va a realizar una prueba con dos tipos de etnias diferentes y que el contrincante del agente elija 1 el 75 % si se trata de una etnia en concreto (mediante *Bernoulli*, como se ha descrito anteriormente).

Las etnias elegidas, a pesar de que existen más, han sido las catalogadas como blancas y latinas. Son dos etnias que pueden llegar a tener rasgos similares en algunos casos y, por lo tanto, puede ser más difícil para el agente llegar a distinguir entre ellas.

En el experimento siguiente será cuando se realice una prueba con más de dos tipos de rasgo diferentes de una misma imagen (como se verá el caso de las emociones, multitud de emociones diferentes, no sólo dos de ellas).

Dentro de la base de datos, al igual que ocurría con el parámetro del género, dentro de los metadatos de cada imagen, existe un campo que es la etnia, dentro de la cual se van a encontrar los dos rasgos a detectar en este experimento, rasgos blancos y rasgos latinos. Estos rasgos están expresados en forma de porcentaje, también como se observó en el anterior experimento. Si al extraer los metadatos de la imagen el porcentaje del rasgo blanco de la cara de una persona es mayor al de latino el contrincante del agente elegirá 1 como decisión. Esto un 75 % de las veces, ya que se quiere, como se ha explicado, acercar el experimento a la realidad, en la que

pueden existir patrones definidos por determinados rasgos de una persona, pero no tienen por que cumplirse el 100 % de las ocasiones.

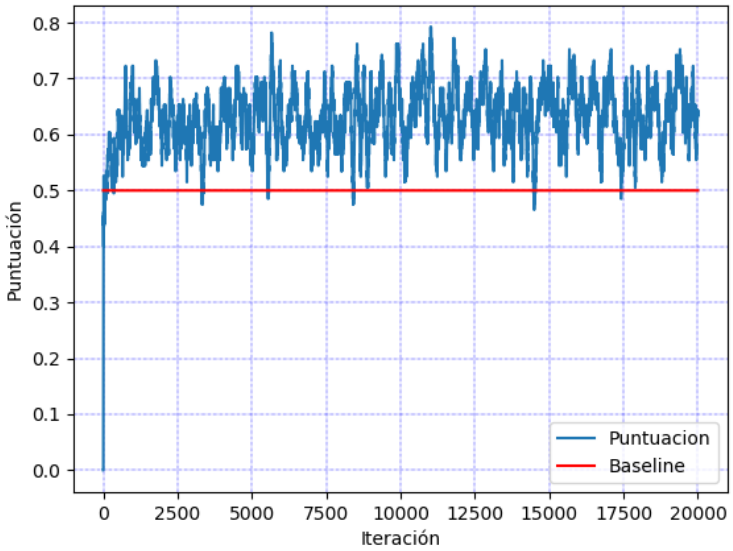


Figura 4.7: Resultados para una decisión en función de la etnia aplicando una probabilidad de Bernoulli de 75 %.

Los resultados de este experimento se encuentran en la figura 4.7. Se pueden analizar los resultados individualmente, sin compararlos con experimentos anteriores. Como se puede observar, la media de puntuación se encuentra en 0.6, es superior al 50 % establecido como *Baseline* y por lo tanto, esto es un factor positivo. El hecho de que no se obtenga una media de puntuaciones superior es comprensible, ya que distinguir en ocasiones entre una persona con rasgos latinos o blancos puede resultar complicado.

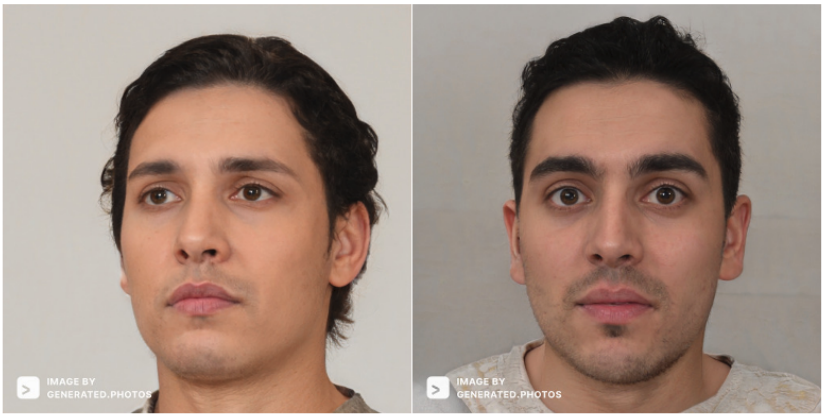


Figura 4.8: Comparación entre imágenes de personas con rasgos latinos y blancos.

En la figura 4.8 se observan dos imágenes extraídas de la base de datos utilizada. A la izquierda se encuentra una persona con rasgos latinos y a la derecha con rasgos blancos. A simple vista puede resultar complicado distinguir entre estos dos rasgos. Los tonos de piel son similares, al igual que el color de ojos y el color del pelo. Es por esto, por lo que al agente le resulta difícil distinguir entre ambos rasgos y los resultados no son tan positivos. El porcentaje que existe en los metadatos que distingue entre rasgos, en casos como este será suficientemente

similar para el caso de la imagen de la derecha que para la de la izquierda, lo que dificulta el proceso de aprendizaje.

Todo esto, unido con la probabilidad de Bernoulli establecida en un 75 %, lo que dificulta, aun más, que se obtengan los mejores resultados posibles.

A pesar de todo esto, no se obtienen resultados negativos, hay que recordar que, el hecho de que la media de puntuación se encuentre por encima de 0.5 (que sería un comportamiento totalmente aleatorio por parte del agente), resulta positivo.

Adicionalmente, se ha realizado otra prueba, omitiendo el factor de la probabilidad de Bernoulli del 75 % que añade una complejidad extra. Es decir, en esta prueba siguiente, para las imágenes con el rasgo "latino"superior al "blanco", el contrincante de la máquina elegirá 0, y 1 en el caso contrario.

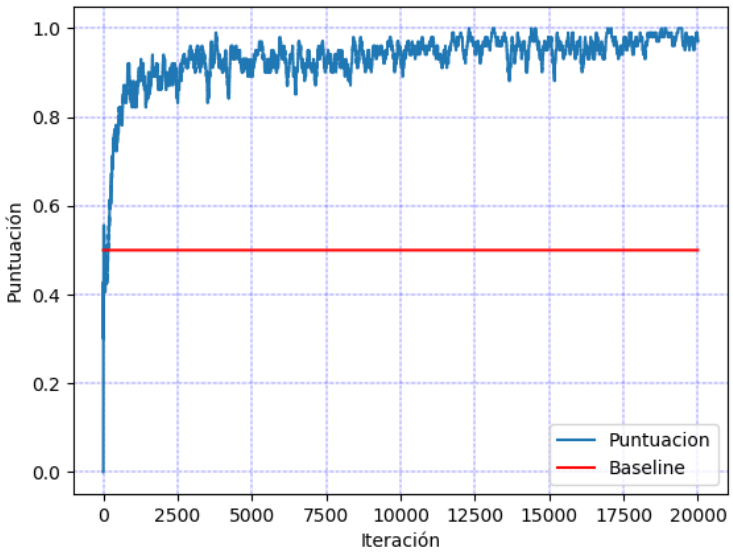


Figura 4.9: Resultados para una decisión en función de la etnia sin probabilidad de Bernoulli.

En la figura 4.9 quedan representados los resultados obtenidos para esta última prueba descrita. Como se ha supuesto, el hecho de suprimir el factor de aplicar una probabilidad de Bernoulli del 75 % mejora los resultados sustancialmente, llegando incluso a una media de 1 en las últimas iteraciones.

4.6. Quinto experimento: Reconocimiento de emociones

Durante experimentos anteriores se han hecho pruebas similares con parámetros como el género o la raza, recientemente. En este caso, este parámetro (emociones) no es binario, como si que lo es el género, y además tiene un gran número de variantes.

Entre las variantes se encuentran la expresión de alegría, tristeza, enfado, miedo, etc... Todas estas emociones se encuentran cuantificadas en los metadatos de las imágenes, al igual que en casos anteriores.

En este caso se parte de que el algoritmo encargado de identificar las emociones en las imágenes ya tiene una complejidad elevada, en comparación con otros parámetros, como la

detección del género, pero se ha comprobado que es capaz de detectar estas emociones de manera correcta, basado en las imágenes de la base de datos.

Las expresiones se detectan en forma de porcentajes, como en anteriores ejemplos, pudiendo existir mezcla de emociones, algo que se aproxima a l mundo real.

Para poder escoger la acción de la maquina en función de la expresión facial, no se puede simplemente elegir una cosa u otra como se hacia con el genero (al ser una decisión binaria). En este caso se han tomado todos los porcentajes de la lista de emociones y se tomara el valor mayor de todos.

Si este valor pertenece a una de las emociones negativas (tristeza, miedo...) se aplicará la función de *Bernoulli* con una probabilidad de 75 % para elegir el valor de la acción de la maquina. Si es neutral se hará con una probabilidad de 10 % y si es positiva con un 15 %.

De esta forma se puede definir un patrón basado en las emociones a la hora de elegir la acción de la máquina. El objetivo es que el agente sea capaz de encontrar este patrón, basado en las emociones, por lo tanto, reconocer emociones en las imágenes y, gracias a este reconocimiento, tomar la decisión que garantice la máxima recompensa posible.

El agente tendrá los mismos parámetros que en pruebas anteriores y se observará el resultado en 50000 iteraciones.

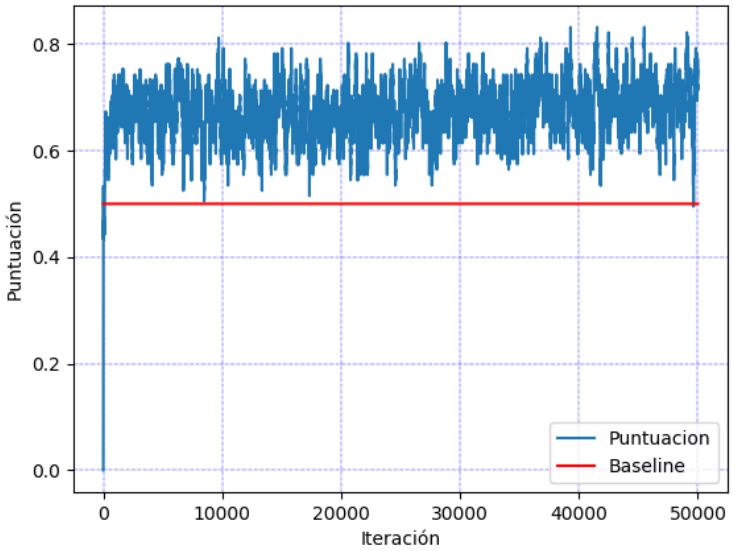


Figura 4.10: Resultados para una decisión en función de las emociones.

Los resultados se pueden observar en la figura 4.10. Como se puede apreciar, estos resultados son positivos. Sin tener una media de puntuaciones elevada, siempre se encuentra por encima de ese umbral de aleatoriedad y por encima incluso de 0.6 sobre 1.

Esto es positivo ya que se ha aumentado sustancialmente la complejidad, añadiendo más parámetros como lo son las emociones y un sistema de decisión de la máquina más complejo también, lo que se traduce en un patrón más difícil de encontrar por parte del agente y por lo tanto en una recompensa obtenida menor de media.

A pesar de esto, la recompensa no es inferior a ese 50 % o en torno a el, y además, se puede observar una ligera tendencia ascendente, por lo tanto, con una mayor cantidad de iteraciones, se podrían incluso mejorar los resultados, eso sí, con un mayor coste computacional y de tiempo (unos factores que no se deben tomar a la ligera en este tipo de simulaciones).

4.7. Sexto experimento: Reglas del dilema del prisionero

Este es el último experimento que se ha realizado y difiere del resto en varios factores.

En primer lugar, el sistema de recompensas cambiará por primera vez, se basará en el dilema del prisionero[14]. Este "dilema"se trata de un problema/teorema perteneciente a la teoría de juegos, el cual plantea que dos personas pueden llegar al caso de no trabajar en equipo, a pesar de que eso sería lo más beneficioso para ambos.

Para representar este problema se han utilizado numerosos ejemplos, ya que se puede aplicar numerosas áreas, pero en el caso de este trabajo, se enfocará en el dilema de trabajar en equipo o de forma solitaria.

	El agente juega en equipo	El agente juega en solitario
La máquina juega en equipo	Ambos obtienen una recompensa de +1 punto	El agente obtiene una recompensa de +2 puntos y la máquina -1 punto
La máquina juega en solitario	La máquina obtiene una recompensa de +2 puntos y el agente-1 punto	Ambos obtienen una recompensa de 0 puntos

Figura 4.11: Reglas del dilema del prisionero.

La forma más sencilla de entender este nuevo sistema de recompensas es mediante la figura 4.11. En ella se puede observar que el caso más beneficioso para ambas partes a largo plazo sería el de jugar siempre en equipo, ya que ambos obtendrían 1 punto de forma constante, sin perder nunca. Lo que nos describe este dilema, aplicado al ser humano, es que existe un miedo de que el contrincante elija jugar en solitario y que le haga perder 1 punto, al estar jugando en equipo. Es por este miedo por el que decide jugar en solitario, influido por el "quedar malz perder puntos.

Este mismo miedo se aplica al contrincante y, por lo tanto, ambos terminan jugando en solitario, algo que, a nivel de recompensa obtenida, resulta irracional.[16]

Al aplicar esto al aprendizaje por refuerzo, el agente, es totalmente racional, lógicamente no existe ese miedo que puede tener un ser humano. Simplemente se busca la máxima recompensa posible a largo plazo.

El sistema de recompensas de la figura 4.11 se ha incorporado directamente al código y es el que tiene el agente en este experimento. Para poder incorporarlo al código y al sistema de decisiones pasado, elegir "1"será elegir jugar solo y "0", en equipo

La máquina va a estar programada para elegir 1 (jugar en solitario) el 70 % de las veces (mediante *Bernoulli*, como anteriormente).

Ante esta situación y observando la tabla del dilema del prisionero, se puede suponer lo que ocurrirá a largo plazo. Y es que el agente (a partir de ciertas iteraciones), elegirá siempre 1, es decir, jugar en solitario. Ya que sólo se pueden dar 2 situaciones, el 70 % de las iteraciones la máquina elegirá 1 también, por lo que ambos ganarán 0, el agente no perderá puntos, mientras que en el 30 % restante, el agente ganará 2 puntos, a la larga, maximizará su recompensa.

De forma adicional, teniendo la elección de la máquina se puede calcular la máxima recompensa posible y observar cuantas iteraciones son necesarias para el agente para alcanzar este resultado, si llega a hacerlo.

Eligiendo la máquina 1 un 70 % de las iteraciones, de 1000 iteraciones, 700 serán 0 puntos y 300, 2 puntos cada una, por lo que serían 600 puntos.

Para esto se ha modificado el programa para que en cada iteración se ejecuten 1000 "partidas", es decir, que en cada iteración se calculen 1000 recompensas para 1000 imágenes al azar.

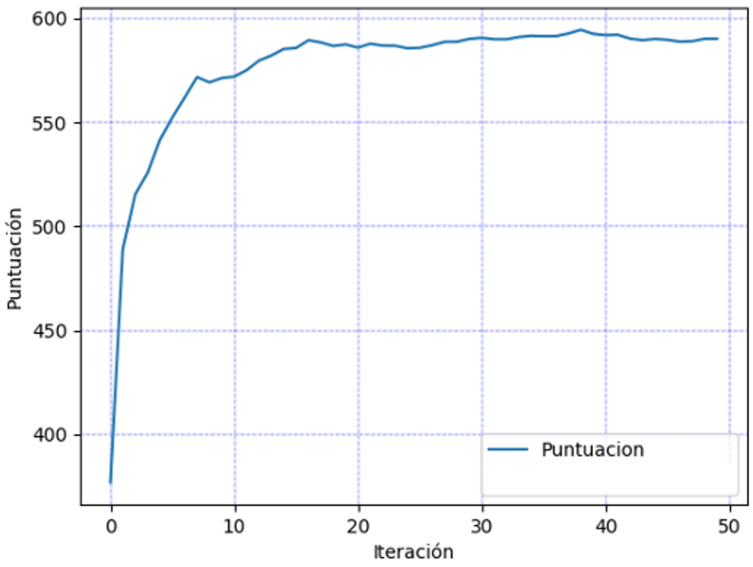


Figura 4.12: Resultado del experimento con el dilema del prisionero.

El resultado se puede observar en la figura 4.12. Se especifican 50 iteraciones, ya que en cada una se calculan 1000 recompensas, por lo que en total serían 50000 recompensa, al igual que en pruebas anteriores.

Como se puede apreciar, se alcanza prácticamente esa recompensa máxima de 600. A pesar de esto, es un proceso lento, ya que hasta la iteración 15-20, no se alcanzan los máximos resultados y se empieza a estabilizar, y hay que recordar que esto serían 15000-20000 ejecuciones con distintas imágenes y recompensas. Esto es algo que se ha visto y comentado anteriormente, el aprendizaje por refuerzo es un algoritmo lento en obtener los mejores resultados pero que como beneficio, precisa de poca interacción humana.

4.8. Conclusiones

A lo largo de todos los experimentos se ha comprobado que la aplicación de algoritmos de aprendizaje por refuerzo resulta positiva, ya que en todo momento, los resultados se encuentran por encima del umbral de la aleatoriedad. Esto es algo que sirve para evaluar si a largo plazo, un sistema así resulta positivo, ya que podría estar aplicado a escenarios en los que los resultados conlleven un desempeño directamente económico.

Se han observado resultados positivos para una serie de rasgos distintos de las imágenes y que, aumentando la complejidad con el establecimiento de patrones y no un comportamiento lineal en el contrincante del agente, este es capaz de alcanzar resultados razonablemente buenos.

Por último, se ha comprobado que, aumentado aun más la complejidad con un sistema de recompensas como el dilema del prisionero, se siguen alcanzando buenos resultados, llegando incluso a los máximos posibles calculados, a partir de un número de iteraciones.

A pesar de esto, algo que ha resultado común a todos los experimentos, es que los algoritmos de aprendizaje por refuerzo son lentos, en comparación con otros algoritmos de aprendizaje supervisado. Son necesarias determinadas iteraciones para alcanzar los mejores resultados posibles. Como parte positiva, como se ha comentado, es la poca interacción humana necesaria, más allá de definir el sistema de recompensas o el número de iteraciones.

5

Conclusiones y Trabajo Futuro

5.1. Conclusiones

Durante esta sección se van a analizar las conclusiones obtenidas en este trabajo de fin de máster y si son acordes a los objetivos planteados al inicio. Para ello se irán recorriendo estos objetivos y analizando si se han podido desarrollar con éxito.

En primer lugar, se ha desarrollado un agente totalmente capaz de ejecutar y aprender mediante algoritmos de aprendizaje por refuerzo. Se ha desarrollado este agente de forma integra y se han estudiado sus parámetros, variándolos, con el fin de obtener los mejores resultados posibles.

Se han realizado pruebas sobre patrones definidos y se ha comprobado que el agente desarrollado es capaz de encontrar estos patrones con la información contenida en los vectores de *embedings*. Todo esto incluso con patrones de una complejidad elevada y que se cumplían sólo en una probabilidad (aplicando *Bernoulli*).

En cuanto a la creación del entorno, se ha podido realizar de forma correcta y añadiendo ese factor más de complejidad mediante un entorno basado en imágenes, con esa captación del vector de *embedings*, que es lo que el propio agente puede observar en cada iteración. La aplicación de este entorno a las imágenes de la base de datos artificial ha sido positiva y gracias a esto, se han podido realizar experimentos con una gran cantidad de iteraciones.

De forma adicional, se han creado dos sistemas de recompensas. Uno más simple y sencillo y otro basado en el dilema del prisionero. Como conclusión se ha podido observar que el sistema de recompensas tiene una gran importancia en los algoritmos de aprendizaje por refuerzo e influye directamente en los resultados

Se ha podido observar que cambiar este sistema de recompensas es fundamental para apreciar nuevos resultados del algoritmo.

Por último, con el fin de entender completamente el comportamiento del agente en un entorno como el de este trabajo (entorno compuesto por imágenes). Se han hecho numerosas pruebas variando todo tipo de parámetros, modo de ejecutar las pruebas, número de iteraciones, etc... Todo esto ha provocado realizar un estudio exhaustivo del agente y de los mejores resultados posibles que puede alcanzar según la prueba realizada.

Se han analizado cada uno de los resultados de forma individual, teniendo en cuenta todas sus variables.

Como conclusión final, afirmar, algo que ya se ha enunciado numerosas veces en este trabajo. Los resultados obtenidos son positivos. Hay casos y experimentos en los que los resultados son extremadamente positivos y otros casos en los que son resultados más austeros, pero en ningún caso se han obtenido resultados que puedan considerarse negativos. Esto es una afirmación a valorar.

También hay que afirmar que, como se ha comentado anteriormente, los algoritmos de aprendizaje por refuerzo son lentos, es decir, son necesarias una cantidad de iteraciones sustancial para alcanzar buenos resultados. Pero también se ha observado que, una vez el agente y el sistema de recompensas se encuentra desarrollado, entre cada una de las pruebas ha sido necesaria poca interacción humana, y mucho menos durante los experimentos. Esto es algo positivo, ya que facilita la labor del programador y proporciona una gran rapidez para realizar experimentos distintos y, así, realizar un estudio mayor de los resultados obtenidos, como se ha podido realizar en este trabajo de fin de máster.

5.2. Trabajo Futuro

En la sección 5.1 se han analizado las conclusiones extraídas de la realización de este trabajo. Durante esta sección se expondrá el trabajo futuro a la realización de todos los experimentos y sistemas que se han desarrollado.

Como se ha comentado, dentro del aprendizaje por refuerzo existen dos ramas fundamentales, aquellos métodos basados en modelos y aquellos que tienen un modelo libre. El caso del método utilizado en este trabajo, el *Q-Learning*, se trata de un método con modelo libre. Dentro del método de *Q-Learning*, se ha utilizado el *DQN*, pero existen más métodos. Como trabajo futuro se podrían comparar los resultados obtenidos con otros métodos dentro del aprendizaje por refuerzo, incluso de una rama distinta.

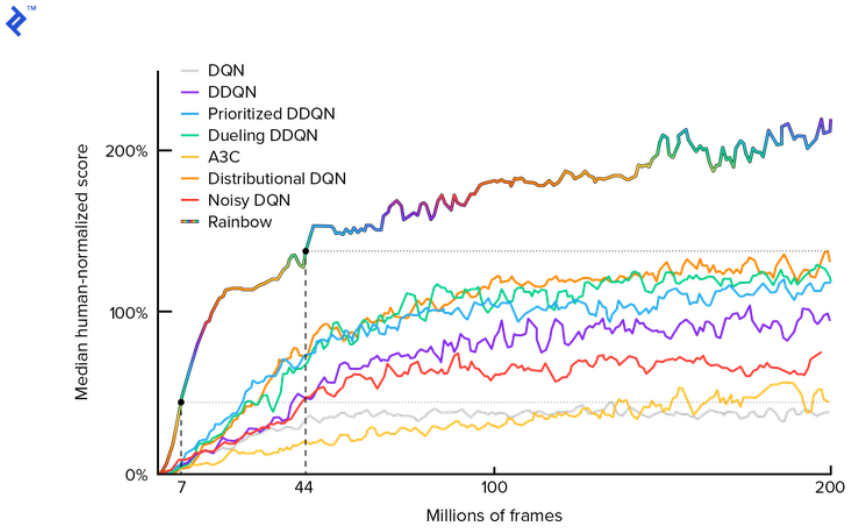


Figura 5.1: Comparación entre distintos algoritmos de *Deep Q-Learning*

De forma adicional, en la figura 5.1 se puede observar una comparación entre distintos algoritmos de *DQN*. Y es que no existe un único algoritmo en este caso, como se puede comprobar,

el campo del aprendizaje por refuerzo tiene una gran profundidad y desarrollo con multitud de algoritmos. La imagen corresponde a un estudio entre los resultados de distintos algoritmos[17]. Como se puede observar, el algoritmo utilizado *DQN* es el que tiene unos resultados más básicos. Se ha utilizado en este trabajo ya que, como primera aproximación a los algoritmos de *DQN* es el más sencillo. Como se puede observar existen algoritmos que proporcionan mejores resultados, siendo el mejor de ellos el denominado como *Rainbow*, que resulta una combinación de todos los demás.

Como trabajo futuro se podría realizar una comparación de todos estos algoritmos con la base de datos utilizada y observar si se cumplen los resultados proporcionados por la figura 5.1. Esto podría suponer una mejora de, como se puede apreciar, incluso más de un 200 % de los resultados obtenidos en este trabajo, algo que merecería la pena analizar.

Adicionalmente, existe un factor importante que puede determinar los resultados y, sobre todo, los recursos necesarios para ejecutar pruebas dentro del aprendizaje por refuerzo (y en cualquier otro campo de la inteligencia artificial). Este factor es el *hardware*.

Todas las pruebas y experimentos que se han realizado, se han ejecutado desde un ordenador portátil personal. Es cierto que hoy en día los componentes de un ordenador moderno y actualizado son más que suficientes para desarrollar y probar sistemas de aprendizaje por refuerzo pero, todo esto depende de la complejidad de los sistemas y del tiempo de entrenamiento establecido.

Se han observado ejemplos de algoritmos como *AlphaGo Zero* que ejecutaban pruebas de aprendizaje por refuerzo durante más de un mes. Esto sería algo inviable en la realización de este trabajo, pero, sin duda, se podrían observar mejores resultados.

Como trabajo futuro cabría la posibilidad de ejecutar las mismas pruebas durante más tiempo de entrenamiento y con componentes más avanzados, lo que reduciría el tiempo de ejecución necesario e incluso se podrían realizar distintas pruebas simultáneamente. Uno de los componentes que más influyen en los resultados es la tarjeta gráfica. Existen multitud de tarjetas gráficas en el mercado, que se actualizan cada cierto tiempo, y que tienen un rendimiento diferente, lo que se traduce en que una tarjeta gráfica más avanzada (acompañada del equipo complementario necesario, procesador, placa base, etc...) podría reducir drásticamente el tiempo de entrenamiento necesario para obtener los mejores resultados posibles.

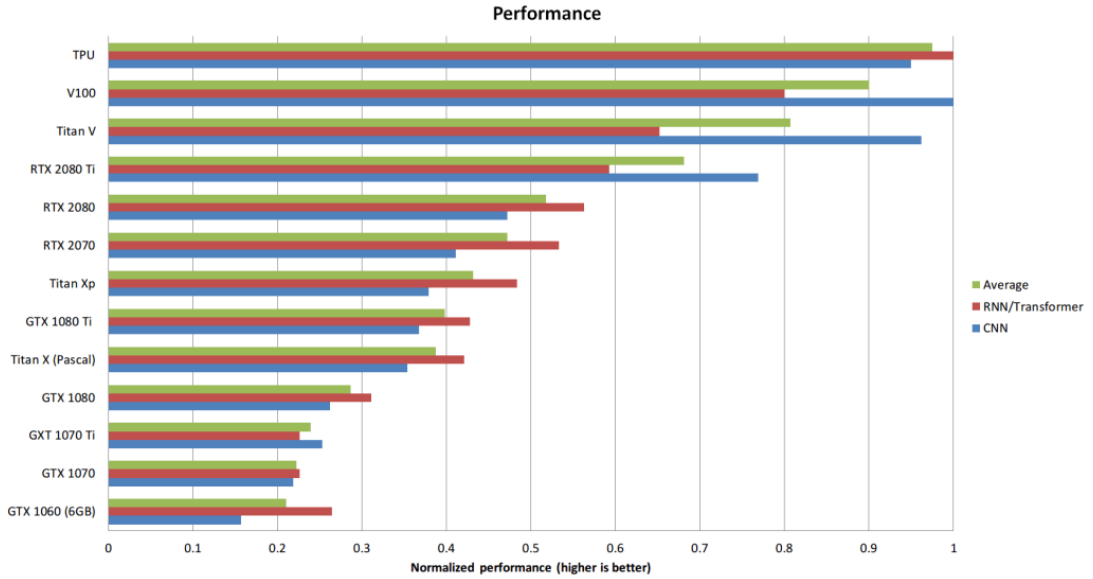


Figura 5.2: Comparación entre rendimientos e distintas GPUs

En la figura 5.2, se puede observar el resultado de un estudio realizado por la popular marca de fabricación de GPUs, *Nvidia*[18]. En el estudio se comparan diferentes modelos de tarjetas gráficas y su rendimiento en populares algoritmos como redes neuronales convolucionales.

Realizando un análisis del rendimiento y del coste de estas tarjetas gráficas se podría adquirir un equipamiento dedicado a realizar los experimentos vistos y mejorar sus resultados.

Por último, se planeta un experimento adicional a los que se han realizado. Este experimento supone un nivel de complejidad significativamente mayor y al rediseño de sistemas como el agente. Se trata de realizar un estudio en el que un agente, no sólo predice la conducta humana mediante algoritmos de aprendizaje por refuerzo, si no que también es capaz de modificar esta conducta. Es decir, el agente, encontrando patrones como se ha estudiado, sería de capaz de realizar decisiones que puedan modificar el comportamiento de la persona humana contra la que esta compitiendo y engañarla, elaborando patrones de decisión, para obtener el máximo beneficio posible.

Esto sería un comportamiento común entre seres humanos, en el que una persona es capaz de engañar a su contrincante, convenciéndole de que va a realizar una acción determinada, cuando, en realidad, va a realizar la acción contraria. Todo esto por su propio beneficio. Este comportamiento se trata de un paso más en los experimentos realizados que podrían suponer un gran avance en relación con este trabajo de fin de máster.

Glosario de acrónimos

- **DQN**: Deep Q-Learning
- **AI**: Artificial Intelligence
- **CNN**: Convolutional Neural Network
- **RL**: Reinforcement Learning
- **ENV**: Enviroment
- **APP**: Application
- **TFM**: Trabajo de Fin de Máster
- **UAM**: Universidad Autónoma de Madrid

Bibliografía

[1] Deepanshu Mehta. State-of-the-art reinforcement learning algorithms. *International Journal of Engineering and Technical Research*, V8, 01 2020.

[2] David Silver. Mastering atari, go, chess and shogi by planning with a learned model. *https://www.nature.com/articles/s41586-020-03051-4*, 2020.

[3] Fernando Sancho Caparrini. Aprendizaje por refuerzo: algoritmo q learning. *http://www.cs.us.es/ fsancho/?e=109*, 2019.

[4] M. Sanjeevi. h:13: Deep reinforcement learning — deep q- learning and policy gradients (towards agi). *https://medium.com/deep-math-machine-learning-ai/ch-13-deep-reinforcement-learning-deep-q-learning-and-policy-gradients-towards-agi-a2a0b611617e*, 2018.

[5] Deepanshu Mehta. State-of-the-art reinforcement learning algorithms. *Obtenido de researchgate:https://www.researchgate.net/publication/338396174*$State - of - the - Art_Reinforcement_Learning_Algorithms$, 2020.

[6] Juan Armando Corbin. John b. watson: vida y obra del psicólogo conductista. *https://psicologiaymente.com/biografias/john-b-watson*, 2020.

[7] Cristian Rusr. Gpt-3, el nuevo modelo de lenguaje de openai, es capaz de programar, diseñar y hasta conversar sobre política o economía. *https://www.xataka.com/robotica-e-ia/gpt-3-nuevo-modelo-lenguaje-openai-capaz-programar-disenar-conversar-politica-economia*, 2020.

[8] Open AI. Entorno gym de openai. `https://gym.openai.com//`.

[9] Satwik Kansal Brendan Martin. Reinforcement q-learning from scratch in python with openai gym. *https://www.learndatasci.com/tutorials/reinforcement-q-learning-scratch-python-openai-gym/*.

[10] David Foster. How to build your own alphazero ai using python and keras. *https://medium.com/applied-data-science/how-to-build-your-own-alphazero-ai-using-python-and-keras-7f664945c188*.

[11] Generated Photos. Base de datos de imágenes sintéticas. `https://generated.photos//`.

[12] Arantxa Herranz. Gdpr/rgpd: qué es y cómo va a cambiar internet la nueva ley de protección de datos. *https://www.xataka.com/legislacion-y-derechos/gdpr-rgpd-que-es-y-como-va-a-cambiar-internet-la-nueva-ley-de-proteccion-de-datos*, 2018.

[13] Paula Rodó. Distribución de bernoulli. *https://economipedia.com/definiciones/distribucion-de-bernoulli.html*, 2020.

[14] Andrés Sevilla Arias. Dilema del prisionero. *https://economipedia.com/definiciones/dilema-del-prisionero.html*, 2015.

[15] Jason Brownlee. A gentle introduction to the rectified linear unit (relu). *https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/l*, 2019.

[16] Alejandro Vera Casas. El dilema del prisionero en la pareja. *https://www.huffingtonpost.es/entry/el-dilema-del-prisionero-en-la-pareja*$_e s_5 e 3 dec 33 c 5 b 6 bb 0 f f c 111036$, 2020.

[17] Adam Stelmaszczyk. A deep dive into reinforcement learning. *https://www.toptal.com/machine-learning/deep-dive-into-reinforcement-learning*, 2018.

[18] Nefi Alarcon. Reinforcement learning algorithm helps train thousands of robots simultaneously. *https://developer.nvidia.com/blog/nvidia-researchers-develop-reinforcement-learning-algorithm-to-train-thousands-of-robots-simultaneously/*, 2018.