



Universidad de Oviedo

ESCUELA POLITÉCNICA DE INGENIERÍA DE GIJÓN

GRADO EN INGENIERÍA INFORMÁTICA EN TECNOLOGÍAS DE LA INFORMACIÓN

ÁREA DE INTELIGENCIA ARTIFICIAL

IA SOMBRA DE AJEDREZ

D. GONZÁLEZ VÁZQUEZ, Juan
TUTOR: D. SÁNCHEZ RAMOS, Luciano

FECHA: JUNIO 2024

Resumen

La inmensa mayoría de las Inteligencias Artificiales (IAs) diseñadas para jugar al ajedrez se crean con un único objetivo: ganar partidas. Han pasado ya muchos años desde que estas IAs superaron la habilidad humana, pero todavía están lejos de ser perfectas. Hoy en día aún se continúa trabajando para que cometan menos fallos, realicen jugadas más brillantes y den menos oportunidades a sus rivales.

El objetivo de este Trabajo Fin de Grado es crear una IA de ajedrez desde un punto de vista diferente. No se busca desarrollar una IA que juegue al más alto nivel, sino una IA capaz de jugar como una persona real, aprendiendo el comportamiento y estrategias de un jugador determinado, de forma que pueda replicar su estilo de juego en partidas con otras personas.

Este enfoque abre la puerta a diversas opciones de juego que serían irrealizables con una IA convencional. Ofrece a los jugadores la posibilidad de compartir su IA con otras personas y les permite jugar contra las de sus amigos o las de otros jugadores, añadiendo así mucha más variedad a las partidas del tipo Jugador VS. CPU.

Palabras clave

Ajedrez, Modelado de jugador, Redes neuronales convolucionales.

Tabla de contenido

1. Introducción	7
1.1. MOTIVACIÓN	7
1.2. AJEDREZ POR COMPUTADOR EN LA ACTUALIDAD	8
1.3. OBJETIVOS Y ALCANCE	12
2. Estudio de la situación actual	16
3. Aspectos teóricos	18
3.1. PORTABLE GAME NOTATION	18
3.2. LEELA CHESS ZERO.....	21
3.3. MAIA CHESS	22
3.5. TRAININGDATA-TOOL.....	25
3.6. BANKSIA GUI	25
4. Trabajo realizado y resultados obtenidos	26
4.1. EQUIPO DE PRUEBAS	26
4.2. OBTENCIÓN Y PROCESAMIENTO DE DATOS	27
4.3. BASE.....	31
4.4. REDUCCIÓN DE PARTIDAS	42
4.5. CAMBIO DE LEARNING RATE	62
4.6. CREACIÓN DE LIBROS DE APERTURAS	76
5. Conclusiones y trabajos futuros	80
Bibliografía	84

Lista de figuras

Figura 1.1 - Interfaz gráfica de ajedrez.....	9
Figura 1.2 - Interfaz de línea de comandos de un motor de ajedrez.....	10
Figura 1.3 - Captura del videojuego Forza Horizon 4.....	13
Figura 1.4 - Captura del videojuego Killer Instinct (2013).....	14
Figura 3.1 - Contenido de un archivo PGN	18
Figura 3.2 - Arquitectura de la red neuronal utilizada por Maia	24
Figura 4.1 - Ejemplo de partida descargada de la base de datos de Lichess	28
Figura 4.2 - Número de partidas por mes del jugador 36.....	35
Figura 4.3 - Número de partidas por mes del jugador 29.....	36
Figura 4.4 - Número de partidas por mes del jugador 21	37
Figura 4.5 - Ventana de creación de libros de aperturas en Banksia GUI.....	78
Figura 4.6 - Ventana de visualización de bases de datos en Banksia GUI.....	79

Lista de tablas

Tabla 4.1 - Componentes hardware del equipo de pruebas.....	27
Tabla 4.2 - Información de los jugadores del conjunto base	32
Tabla 4.3 - Configuración por defecto del learning rate.....	34
Tabla 4.4 – Mejora de precisión (“policy” y “value”) del conjunto base respecto a Maia-1900 en los pasos de entrenamiento 100, 500 y 1000	41
Tabla 4.5 - Información de los jugadores del conjunto 1000 (solo se tienen en cuenta datos del conjunto de entrenamiento)	44
Tabla 4.6 - Información de los jugadores del conjunto 500 (solo se tienen en cuenta datos del conjunto de entrenamiento)	45
Tabla 4.7 - Información de los jugadores del conjunto 250 (solo se tienen en cuenta datos del conjunto de entrenamiento)	46
Tabla 4.8 - Información de los jugadores del conjunto 100 (solo se tienen en cuenta datos del conjunto de entrenamiento)	47
Tabla 4.9 - Información de los jugadores del conjunto 50 (solo se tienen en cuenta datos del conjunto de entrenamiento)	48
Tabla 4.10 - Mejora de precisión (“policy” y “value”) del conjunto 1000 respecto a Maia-1900 en los pasos de entrenamiento 100, 500 y 1000	52
Tabla 4.11 - Mejora de precisión (“policy” y “value”) del conjunto 500 respecto a Maia-1900 en los pasos de entrenamiento 100, 500 y 1000	54
Tabla 4.12 - Mejora de precisión (“policy” y “value”) del conjunto 250 respecto a Maia-1900 en los pasos de entrenamiento 100, 500 y 1000	56
Tabla 4.13 - Mejora de precisión (“policy” y “value”) del conjunto 100 respecto a Maia-1900 en los pasos de entrenamiento 100, 500 y 1000	58
Tabla 4.14 - Mejora de precisión (“policy” y “value”) del conjunto 50 respecto a Maia-1900 en los pasos de entrenamiento 100, 500 y 1000	60
Tabla 4.15 - Nuevo learning rate, ajustado para utilizarse con 1000 pasos de entrenamiento	63
Tabla 4.16 - Mejora de precisión (“policy” y “value”) del conjunto 1000 respecto a Maia-1900 en los pasos de entrenamiento 100, 500 y 1000 utilizando el nuevo learning rate	66

Tabla 4.17 - Mejora de precisión (“policy” y “value”) del conjunto 500 respecto a Maia-1900 en los pasos de entrenamiento 100, 500 y 1000 utilizando el nuevo learning rate 68

Tabla 4.18 - Mejora de precisión (“policy” y “value”) del conjunto 250 respecto a Maia-1900 en los pasos de entrenamiento 100, 500 y 1000 utilizando el nuevo learning rate 70

Tabla 4.19 - Mejora de precisión (“policy” y “value”) del conjunto 100 respecto a Maia-1900 en los pasos de entrenamiento 100, 500 y 1000 utilizando el nuevo learning rate 72

Tabla 4.20 - Mejora de precisión (“policy” y “value”) del conjunto 50 respecto a Maia-1900 en los pasos de entrenamiento 100, 500 y 1000 utilizando el nuevo learning rate 74

1. Introducción

1.1. MOTIVACIÓN

En el campo de la Inteligencia Artificial siempre se ha mostrado un especial interés en el desarrollo de algoritmos capaces de jugar al ajedrez. Esta inclinación por el ajedrez, y no por otros juegos, se debe principalmente a su complejidad. El elevado número de posibles posiciones (aproximadamente 10^{44}) y la gran extensión de su árbol de juego, es decir, las diferentes formas en las que una partida puede resolverse (10^{123}) [1], hacen de él el campo de pruebas perfecto para el desarrollo y la evaluación de algoritmos que, además de emplearse para jugar al ajedrez, también pueden ser utilizados en otras aplicaciones.

De esta relación entre la Inteligencia Artificial y el ajedrez surgieron los primeros programas capaces de jugar, aunque a muy bajo nivel, partidas completas. Fue necesario el paso del tiempo, con los subsecuentes avances en hardware y software, para que estos programas pudieran derrotar a jugadores experimentados. En el año 1996 se alcanza un punto de inflexión con la famosa victoria de DeepBlue, un computador de IBM diseñado para jugar al ajedrez, ante Garry Kasparov, el por entonces campeón mundial.

Hoy en día, 28 años después de la histórica victoria de DeepBlue, las IAs de ajedrez poseen un nivel de juego sobrehumano, hasta tal punto que ninguna persona es capaz de plantarles cara. Aun así, sus desarrolladores continúan trabajando y compitiendo entre sí con el objetivo de crear la mejor IA de ajedrez posible.

Sin embargo, este enfoque ignora un aspecto muy importante: al tratar al ajedrez únicamente como un problema a resolver, deja de lado el hecho de que este es un juego, y que la gran mayoría de la gente que lo juega lo hace por diversión. Si bien es cierto que las IAs de ajedrez permiten al usuario elegir un nivel de dificultad que se adecúe a sus habilidades, las jugadas que estas realizan no son acordes con las que haría una persona real, realizando movimientos más bien torpes en dificultades bajas y obligando a sus contrincantes a usar estrategias específicas contra ordenadores en dificultades altas.

En conclusión, las IAs de ajedrez convencionales no juegan como lo haría una persona real, sino que siempre reaccionan de la manera óptima ante los movimientos de su oponente (acorde a su dificultad), lo que hace que las partidas contra ellas se vuelvan monótonas e incluso frustrantes. Para solucionar este problema, en este trabajo se propone acercar el modo Jugador vs. CPU al modo Jugador vs. Jugador. Es decir, que de la misma manera que los jugadores pueden buscar partidas en línea contra diferentes personas, las cuales tienen su propio estilo de juego, también deberían poder jugar contra diferentes IAs, cada una con su propio comportamiento.

Para alcanzar esta meta, en este trabajo se desarrollará una IA individualizada, capaz de aprender el estilo de juego de una persona específica a partir de sus partidas con el fin de replicarlo en futuros enfrentamientos. De esta forma, los usuarios tendrían la posibilidad de entrenar a su propia IA para que juegue como ellos y compartirla con la comunidad, formando así una amplia colección de IAs de ajedrez generadas por los jugadores.

1.2. AJEDREZ POR COMPUTADOR EN LA ACTUALIDAD

Para poder comprender el objetivo y el alcance de este trabajo, primero es necesario entender cómo se juega al ajedrez en equipos informáticos en la actualidad.

Independientemente de si se quiere jugar en una página web o localmente en el equipo del usuario, lo más habitual es hacerlo mediante una Interfaz Gráfica de Usuario (GUI). El objetivo principal de este programa es presentar al usuario la información de la partida y permitirle realizar movimientos de manera sencilla [2]. Como se puede ver en la figura 1.1, la interfaz muestra una representación gráfica del tablero, sobre el cual el usuario puede realizar movimientos clicando y arrastrando sus piezas a la posición deseada. Además, también muestra otros datos relevantes como el tiempo restante de cada jugador, los movimientos realizados, etc.



Figura 1.1 - Interfaz gráfica de ajedrez

Sin embargo, si se quiere jugar contra una IA de ajedrez, no basta con disponer de una interfaz gráfica, sino que también es necesario utilizar un motor de ajedrez. El significado de motor de ajedrez ha ido cambiando a lo largo de los años, pero una definición actual podría ser la siguiente: un programa de backend con una interfaz de línea de comandos, el cual analiza las diferentes posiciones del tablero con el fin de realizar el movimiento que genere más posibilidades de ganar la partida.

No es estrictamente necesario utilizar una interfaz gráfica cuando se hace uso de un motor de ajedrez, pero es muy habitual combinar ambos programas debido a que la interfaz de línea de comandos del motor solo permite la comunicación por medio de cadenas de texto. Su uso conjunto es tan común que existen varios protocolos de comunicación entre ambos programas, siendo el protocolo UCI (Universal Chess Interface) el más utilizado.

```

[ - ]
[ - ] v0.28.2 built Sep 23 2022
go nodes 10
Found pb network file: ./1700-0-60-30000.pb.gz
Creating backend [cudnn-auto]...
Switching to [cudnn]...
CUDA Runtime version: 11.8.0
WARNING: CUDA Runtime version mismatch, was compiled with version 11.7.0
Cudnn version: 8.5.0
Latest version of CUDA supported by the driver: 11.8.0
GPU: NVIDIA GeForce GTX 1080
GPU memory: 7.92633 GiB
GPU clock frequency: 1733.5 MHz
GPU compute capability: 6.1
info depth 1 seldepth 2 time 725 nodes 2 score cp 20 nps 2000 tbhits 0 pv d2d4 d7d5
info depth 2 seldepth 3 time 726 nodes 3 score cp 17 nps 1500 tbhits 0 pv d2d4 d7d5 g1f3
info depth 2 seldepth 4 time 727 nodes 4 score cp 13 nps 1333 tbhits 0 pv d2d4 d7d5 g1f3 g8f6
info depth 3 seldepth 5 time 729 nodes 5 score cp 14 nps 1250 tbhits 0 pv d2d4 d7d5 g1f3 g8f6 g2g3
info depth 3 seldepth 6 time 730 nodes 6 score cp 12 nps 1200 tbhits 0 pv d2d4 d7d5 g1f3 g8f6 g2g3 e7e6
info depth 4 seldepth 7 time 731 nodes 7 score cp 14 nps 1166 tbhits 0 pv d2d4 d7d5 g1f3 g8f6 g2g3 e7e6 f1g2
info depth 4 seldepth 8 time 732 nodes 8 score cp 12 nps 1142 tbhits 0 pv d2d4 d7d5 g1f3 g8f6 g2g3 e7e6 f1g2 f8e7
info depth 5 seldepth 9 time 733 nodes 9 score cp 15 nps 1125 tbhits 0 pv d2d4 d7d5 g1f3 g8f6 g2g3 e7e6 f1g2 f8e7 e1g1
info depth 5 seldepth 10 time 734 nodes 10 score cp 12 nps 1111 tbhits 0 pv d2d4 d7d5 g1f3 g8f6 g2g3 e7e6 f1g2 f8e7 e1g1 e8g8
bestmove d2d4 ponder d7d5

```

Figura 1.2 - Interfaz de línea de comandos de un motor de ajedrez

Es importante destacar que un motor de ajedrez no es tan solo una IA de ajedrez, sino que también posee otros elementos como la interfaz y el protocolo de comunicación. Los componentes del motor encargados de la IA, es decir, del análisis de las diferentes posiciones del tablero y la elección de un movimiento, son la función de evaluación y la función de búsqueda [3].

La función de evaluación se encarga de estimar lo buena que es una posición dentro del árbol de juego [4], es decir, analiza cómo quedaría el tablero tras realizar un determinado movimiento y devuelve un valor, normalmente numérico, indicando el beneficio de llevar a cabo dicha acción. Existen diferentes maneras de implementar una función de evaluación, pero las siguientes son las más comunes:

- **A mano:** en los primeros motores de ajedrez y en muchos de la actualidad, la función de evaluación está formada por una serie de reglas heurísticas introducidas por el programador del sistema con la ayuda de jugadores experimentados. Estas reglas tienen en cuenta diferentes factores para determinar cuan bueno es un movimiento, como las piezas que están en el tablero, que zonas están siendo protegidas y atacadas, la seguridad del rey, etc.
- **Redes neuronales:** con el auge del Deep Learning o Aprendizaje profundo en los últimos años, algunos motores de ajedrez empezaron a utilizar redes neuronales para implementar su función de evaluación. A diferencia del enfoque anterior, los

parámetros que determinan la idoneidad de un movimiento no son elegidos por el programador, sino que son descubiertos por la propia red durante su entrenamiento.

Como ya se ha dicho, la función de evaluación tan solo analiza una posición del árbol de juego. Esto supone un problema, ya que impide realizar jugadas estratégicas compuestas por varios movimientos y prever las realizadas por los oponentes. Por este motivo y debido a que la complejidad del árbol de juego hace que sea imposible analizar cada una de sus posiciones, surge la necesidad de utilizar una función de búsqueda. El objetivo de dicha función es analizar, utilizando la función de evaluación, el árbol de juego y descartar las ramas, es decir, las combinaciones de movimientos, que conlleven una probabilidad menor de ganar la partida [5]. De esta forma la función es capaz de elegir el movimiento, teniendo en cuenta las amenazas y oportunidades futuras, que genere la mayor probabilidad de derrotar a su oponente. Existen varios algoritmos para implementar esta función, pero los más utilizados son “alpha-beta search” y “Monte Carlo tree search”.

A pesar de todo lo dicho anteriormente, la función de evaluación y la función de búsqueda solo suelen usarse durante la etapa de medio juego, empleándose otros métodos para las fases de apertura y final.

La fase de apertura es la primera etapa de la partida, suele abarcar las 10 primeras jugadas y es en la que los jugadores comienzan a desarrollar sus piezas partiendo de la posición inicial. Esta fase supone un problema a las IAs de ajedrez cuando juegan con blancas, ya que al tener que jugar el primer movimiento de la partida siempre terminan realizando el mismo (ante la misma posición el mejor movimiento siempre es el mismo), lo que hace que las partidas pierdan dinamismo. Para resolver este problema se inventaron los libros de apertura, que no son más que ficheros que contienen una serie de aperturas diferentes, es decir, distintos conjuntos de movimientos con los que iniciar la partida. De esta forma, la IA puede elegir una apertura del libro con un determinado grado de aleatoriedad, evitando así realizar siempre las mismas jugadas [6].

Tampoco es habitual utilizar las funciones de evaluación y búsqueda durante la fase final, aunque por motivos diferentes. La fase final termina con el desenlace de la partida, pero no

tiene un comienzo bien definido, se dice que empieza cuando el número de piezas en el tablero es reducido. Que haya un escaso número de piezas significa que es posible utilizar una base de datos con la probabilidad de victoria precalculada para cada posición del árbol de juego, evitando la necesidad de utilizar las funciones de evaluación y búsqueda para calcular dicha probabilidad en tiempo de ejecución. Estas bases de datos son conocidas como “endgame tablebases” [7], siendo las “Syzygy tablebases” las más populares en la actualidad.

Es importante destacar que, debido al protocolo UCI, los libros de apertura por defecto no forman parte del motor de ajedrez, sino que son archivos independientes ejecutados por la interfaz gráfica. De la misma manera, las “endgame tablebases” también pueden ser utilizadas por la interfaz [8].

En conclusión, una inteligencia artificial de ajedrez está dividida en varias partes (incluso en varios programas) cada una de las cuales puede implementarse de múltiples formas. Sin embargo, hay dos que marcan claramente el estilo de juego de la IA. Sin duda, la más influyente en su forma de jugar es la función de evaluación, ya que es la encargada de decidir cuál es el mejor movimiento en cada situación. No obstante, el libro de apertura también es importante, ya que en él se encuentran todas las aperturas que la IA puede llegar a jugar.

1.3. OBJETIVOS Y ALCANCE

Como se ha comentado en el apartado “1.1. Motivación”, el objeto de este trabajo es crear una IA de ajedrez que juegue como una persona real, aprendiendo el estilo y las estrategias de un jugador determinado a partir de sus partidas, de forma que pueda replicar su comportamiento en futuros enfrentamientos.

Este concepto de IA individualizada es relativamente reciente en el ámbito del ajedrez, pero no es una novedad en el mundo de los videojuegos. De hecho, existen varias aplicaciones de este tipo, como Drivatar o Shadow AI (de donde toma el nombre este trabajo), las cuales llevan años utilizándose con éxito en sus respectivos juegos con los mismos fines que se

plantean en este documento. Es por esta razón que, antes de definir en detalle los objetivos y el alcance, es necesario estudiar estas aplicaciones para encontrar las claves de su éxito.

Drivatar es una de las implementaciones de aprendizaje automático más longevas de la industria de los videojuegos, siendo utilizada en todas las entregas de la serie de carreras Forza desde su debut en el año 2005. Para conseguir replicar el estilo de conducción de un jugador determinado, Drivatar utiliza una red neuronal la cual es entrenada con las partidas del jugador que quiere imitar. De esta forma, la IA es capaz de inferir el comportamiento del jugador ante cualquier situación, incluso si este todavía no la ha experimentado (diferentes coches, circuitos, condiciones meteorológicas, etc.) [9].



Figura 1.3 - Captura del videojuego Forza Horizon 4

Shadow AI es más reciente, siendo introducida en 2015 en el reboot de 2013 del clásico juego de lucha Killer Instinct. La diferencia principal con Drivatar es que esta no hace uso de una red neuronal para replicar el comportamiento del jugador, sino que utiliza una técnica de inteligencia artificial conocida como Case-Based Reasoning (CBR). Gracias a esta técnica, Shadow AI es capaz de extraer y almacenar los patrones de comportamiento que muestra un jugador ante una determinada situación, de forma que dada una situación similar, esta pueda acceder a ellos, adaptarlos y utilizarlos para afrontarla [10].

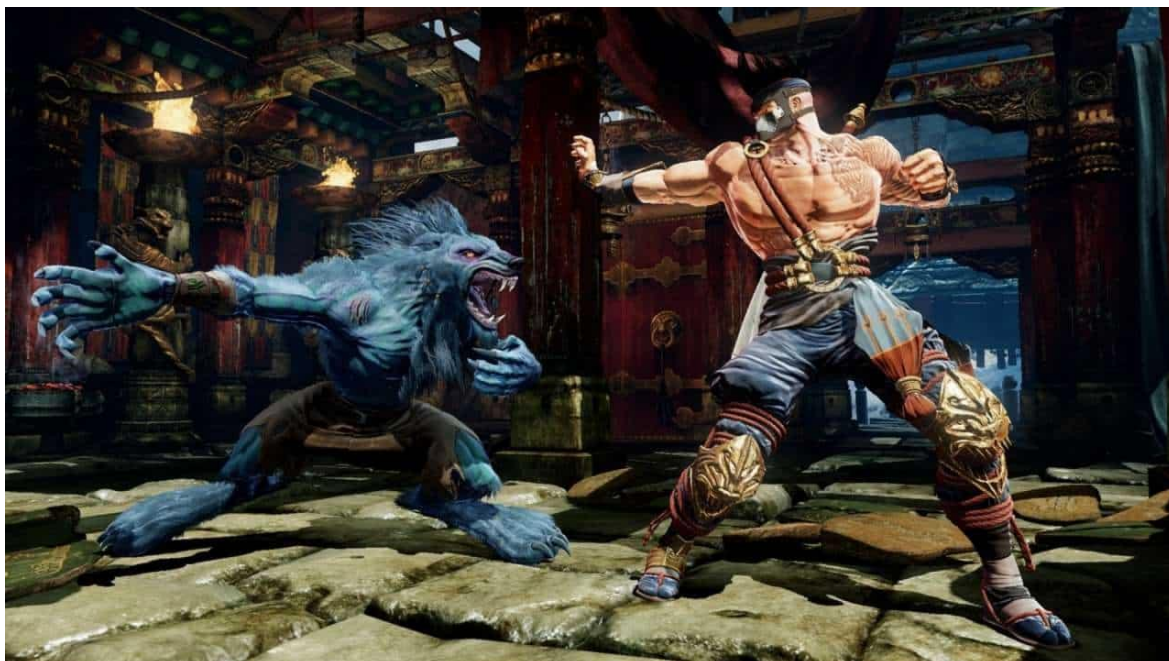


Figura 1.4 - Captura del videojuego Killer Instinct (2013)

A pesar de sus diferencias y de pertenecer a franquicias tan distintas, al final tanto Drivatar como Shadow AI buscan lo mismo: aprender a jugar a su respectivo juego con el estilo de un jugador determinado para añadir variedad a sus modos de un jugador. Para alcanzar este objetivo, ambas se basan en los siguientes principios:

- **Funcionamiento con muy pocos datos:** antes de que la IA aprenda el comportamiento de un jugador ha de pasar por una fase de obtención de datos, en la cual el jugador ha de completar un número de partidas suficiente para poder extraer de ellas su estilo de juego. Es importante que esta fase dure lo menos posible, incluso si eso significa reducir la calidad de las predicciones de la IA, ya que si se alarga demasiado las probabilidades de que el jugador pierda el interés aumentarían significativamente. Por ejemplo, para conseguir que Drivatar sea funcional, el jugador ha de realizar tres vueltas a cinco circuitos diferentes, mientras que para poder empezar a utilizar Shadow AI tan solo debe completar tres combates.
- **Actualización con nuevas partidas:** existen dos razones por las que es necesario actualizar la IA con nuevas partidas. La primera surge a raíz del punto anterior, ya que en el momento en el que se crea la IA esta tiene muy pocos datos y es necesario añadir más para conseguir una mayor calidad en las predicciones. La segunda se debe

a que el estilo de un jugador no es estático, sino que según pasa el tiempo este va mejorando y aprendiendo nuevas estrategias, las cuales deben aparecer reflejadas en las decisiones de la IA.

- **Tiempo de aprendizaje muy reducido:** el tiempo que invierte la IA en asimilar el estilo de juego de un jugador a partir de sus partidas ha de ser lo más reducido posible, a pesar de que esto signifique empeorar la calidad de sus predicciones. Al igual que en el primer punto, el tiempo del jugador es lo más importante.

Siguiendo el ejemplo de Drivatar y Shadow AI, la IA propuesta en este trabajo también debe basarse en estos conceptos. Por tanto, el objeto de este trabajo no es solo crear una IA que juegue como un jugador determinado, sino que además cumpla con los tres puntos anteriores.

2. Estudio de la situación actual

En la actualidad no existe ninguna IA de ajedrez que cumpla todos los objetivos propuestos en este documento. Sin embargo, sí que hay sistemas libres de código abierto que satisfacen varios y que, por tanto, pueden utilizarse como base para llevar a cabo este trabajo.

Como ya se ha explicado en el apartado “1.2. Ajedrez por computador en la actualidad”, los componentes fundamentales de toda inteligencia artificial de ajedrez se encuentran en el motor de ajedrez. Por esta razón, es necesario encontrar un motor que permita utilizar las partidas jugadas como fuente de conocimiento para su función de evaluación.

Para que una función de evaluación pueda “aprender” a partir de una serie de partidas, esta ha de estar implementada con un algoritmo que lo permita, como por ejemplo CBR o una red neuronal. En el caso de CBR no existe ningún motor de ajedrez que lo utilice, lo cual puede deberse a la gran complejidad técnica que implica su implementación. Sin embargo, con la popularidad del Deep Learning en los últimos años, las redes neuronales han empezado a ganar terreno a los algoritmos más clásicos, llegando al punto de igualar e incluso superar en fuerza a los motores tradicionales más potentes.

Leela Chess Zero es el ejemplo más notable del enfoque anterior. Leela surge en el año 2018 inspirado por el proyecto de Google “AlphaZero” como un motor de ajedrez libre y de código abierto, el cual hace uso de una red neuronal convolucional (CNN) para evaluar las diferentes posiciones del árbol de juego con el fin de elegir el mejor movimiento posible. Para conseguir que dicha red aprenda a jugar al ajedrez, sus desarrolladores la entrenan con partidas generadas a partir de “self-play” (es decir, partidas que la IA juega contra sí misma), y la van mejorando continuamente gracias a una técnica conocida como “Reinforcement Learning”. Gracias a estos métodos, Leela se ha convertido en uno de los motores de ajedrez más fuertes del mundo, llegando a ganar a Stockfish, el motor de ajedrez más potente en la actualidad, en varios torneos [11].

Las cualidades de Leela, es decir, que sea un motor de ajedrez libre, de código abierto y que utilice una red neuronal para implementar su función de evaluación, lo convierten en el motor idóneo sobre el cual desarrollar este trabajo. El hecho de que su función de evaluación esté compuesta únicamente por una red neuronal significa que si esta es entrenada con las partidas de un jugador determinado en vez de con partidas generadas a partir de “self-play”, la red aprendería a jugar al ajedrez con el estilo de dicho jugador.

La idea de entrenar esta red con las partidas de un jugador determinado no es ninguna novedad, sino que en la actualidad hay un proyecto de investigación en curso que está trabajando en ello. El nombre de este proyecto es Maia Chess, y su objetivo principal es crear una IA que realice los movimientos que haría un jugador humano [12] [13]. A pesar de que esto suene muy parecido a lo que se intenta conseguir en este documento, existen diferencias importantes entre ambos:

- **Tiempo de entrenamiento:** el tiempo de entrenamiento de la red no es un factor importante que Maia tenga en cuenta, por lo que este es elevado. Sin embargo, para este trabajo es un elemento fundamental.
- **Número de partidas necesario para el entrenamiento:** para que Maia pueda funcionar correctamente necesita que la red se entrene con un número elevado de partidas. Sin embargo, para este trabajo el número de partidas ha de ser muy reducido.

Maia está formado por varios programas, pero el más interesante desde el punto de vista de este trabajo es la modificación de la “training pipeline” de Leela. Este es un programa utilizado por los desarrolladores de Leela para entrenar su red neuronal, pero que ha sido modificado por los creadores de Maia para que la IA realice movimientos más humanos. Gracias a que el software de Maia es libre y de código abierto, en este trabajo se utilizará dicho programa de manera conjunta con Leela como base para su desarrollo.

3. Aspectos teóricos

En este capítulo se abordarán las diferentes tecnologías utilizadas a lo largo de este trabajo.

3.1. PORTABLE GAME NOTATION

Portable Game Notation, mejor conocido como PGN, es un estándar para la representación de las partidas de ajedrez basado en archivos de texto ASCII con el objetivo de facilitar el intercambio de información de las partidas entre jugadores, editores e investigadores de cualquier parte del mundo [14].

En esta sección se abordarán los aspectos básicos de PGN con el fin de entender este estándar, puesto que las partidas que se utilizarán posteriormente para entrenar la red neuronal de Leela están almacenadas en archivos de este tipo.

La siguiente figura es un ejemplo sencillo de un archivo PGN.

```
[Event "Rated Blitz game"]
[Site "https://lichess.org/5nT5Vvf2"]
[Date "?????.???.??"]
[Round "?"]
[White "coralines"]
[Black "jlf5"]
[Result "0-1"]

1. d4 d5 2. Nf3 e6 3. g3 Bd6 4. Bg2 f5 5. O-O Nf6 6. Ne5 Bxe5 7. dxe5 Ne4
8. f3 Nc5 9. b4 Nca6 10. a3 O-O 11. f4 b5 12. Be3 c6 13. c3 c5 14. bxc5 Qc7
15. Qb3 Nxc5 16. Qxb5 Ne4 17. Qb2 a5 18. Bd4 Ba6 19. Re1 Bc4 20. Nd2 Na6
21. Nxc4 Qxc4 22. e3 Rab8 23. Qc2 Rb3 24. Bf1 Qa4 25. Bxa6 Rfb8 26. Ra2 Qc6
27. Bd3 a4 28. Rc1 Qe8 29. c4 Rc8 30. c5 Qh5 31. Bxe4 fxe4 32. Qe2 Qh3 33.
Qg2 Qg4 34. h3 Qg6 35. g4 Rd3 36. Rb1 h6 37. Kh2 Kh7 38. Rb7 Qe8 39. f5 Qc6
40. Rab2 exf5 41. gxf5 Rg8 42. e6 Qe8 43. h4 Rxa3 44. h5 Rd3 45. Qg6+ Qxg6
46. fxg6+ Kh8 47. e7 a3 48. Ra2 Re8 49. Rd7 Kg8 50. Rxd5 Rb3 51. Rd8 Rb8
52. Rxe8+ Rxe8 53. Rxa3 Rc8 0-1
```

Figura 3.1 - Contenido de un archivo PGN

Como se puede ver, el archivo está dividido en dos secciones: la sección de etiquetas y la sección de movimientos.

La sección de etiquetas puede estar compuesta por cero o más etiquetas y siempre se encuentra al principio del archivo. En este ejemplo, la sección está formada por siete etiquetas, las cuales reciben el nombre de STR (Seven Tag Roster) y deben aparecer en todas las partidas que se vayan a archivar. El significado de estas etiquetas es el siguiente:

- **Event:** tipo de partida. En este ejemplo es una partida competitiva del modo Blitz, pero podría ser el nombre de un torneo.
- **Site:** lugar en el que se juega la partida. En este ejemplo es la página web de lichess, pero podría ser el nombre de una ciudad.
- **Date:** fecha en la que se juega la partida. En este ejemplo la fecha es desconocida.
- **Round:** ronda (normalmente de un torneo) en la que se encuentra la partida. En este ejemplo la ronda es desconocida.
- **White:** nombre del jugador que juega como blancas.
- **Black:** nombre del jugador que juega como negras.
- **Result:** resultado final de la partida. Este puede ser “1-0” en caso de que ganen las blancas, “0-1” en caso de que ganen las negras, “1/2-1/2” en caso de tablas, o “*” en caso de que el resultado no sea conocido (partida en curso, abandonada, etc.).

En la segunda mitad del archivo, separada por una línea en blanco de la sección de etiquetas, se encuentra la sección de movimientos, la cual está formada por todos los movimientos de la partida. Como se puede ver en la figura, esta sección finaliza con el resultado de la partida y está formada por múltiples cadenas de texto con el siguiente formato:

<Número movimiento>. <Movimiento blancas> <Movimiento negras>

El significado de “Número movimiento” es bastante intuitivo, simplemente indica el número del movimiento realizado por las blancas y por las negras. Sin embargo, entender el significado de dichos movimientos es una tarea más compleja. Estos siguen la notación SAN (Standard Algebraic Notation), lo que significa que están formados por el tipo de la pieza que se mueve y la casilla del tablero a donde esta se desplaza.

Cada tipo de pieza es codificado con una letra que lo representa:

- **Peón:** “P”, aunque normalmente no se utiliza ningún carácter.
- **Caballo:** “N”.
- **Alfil:** “B”.
- **Torre:** “R”.
- **Reina:** “Q”.
- **Rey:** “K”.

Cada movimiento es codificado con una combinación de su fila, representada con un número del 1 al 8, y su columna, representada con una letra de la “a” a la “h”.

Conociendo estos conceptos, se puede saber que el primer movimiento de las blancas (d4) significa peón a la casilla d4 y el primer movimiento de las negras (d5) significa peón a la casilla d5. Sin embargo, en la figura se puede ver que aparecen caracteres que no corresponden con ninguna pieza ni con ningún movimiento. Esto se debe a que son utilizados para indicar las siguientes situaciones en la partida:

- **Captura de pieza:** “x”. Se añade como prefijo a la casilla de destino. Por ejemplo, el movimiento “Bxe5” significa que el alfil captura la pieza que se encuentra en la casilla e5.
- **Enroque corto:** “O-O”.
- **Enroque largo:** “O-O-O”.
- **Promoción de peón:** “=”. Es inmediatamente seguido por el tipo de pieza al que promociona el peón. Por ejemplo, el movimiento “e8=Q” significa que el peón se mueve a la casilla e8 y promociona a una reina.
- **Jaque:** “+”. Se añade como sufijo al movimiento. Por ejemplo, el movimiento “Qg6+” significa que la reina se mueve a la casilla g6 y hace jaque al rey.
- **Jaque mate:** “#”. Se añade como sufijo al movimiento. Por ejemplo, el movimiento “Rf8#” significa que la torre se mueve a la casilla f8 y hace jaque mate al rey.

Por último, hay que tener en cuenta que realizar la codificación de un movimiento únicamente con el tipo de pieza que se mueve y la casilla de destino de dicha pieza puede ser ambiguo en algunas situaciones, ya que puede darse el caso de que más de una pieza del

mismo tipo pueda ejecutar dicho movimiento. Para solucionar este problema las piezas se pueden diferenciar de las siguientes formas:

1. **Por la columna de su casilla de inicio:** por ejemplo, el movimiento “Nfd4” significa que el caballo que se encuentra en la columna “f” se mueve a la casilla “d4”.
2. **Por la fila de su casilla de inicio:** por ejemplo, el movimiento “N3d4” significa que el caballo que se encuentra en la fila “3” se mueve a la casilla “d4”.
3. **Por la fila y la columna de inicio:** por ejemplo, el movimiento “Nf3d4” significa que el caballo que se encuentra en la casilla “f3” se mueve a la casilla “d4”.

3.2. LEELA CHESS ZERO

En el capítulo “2. Estudio de la situación actual” ya se ha dado una pequeña explicación de este motor de ajedrez. Sin embargo, es necesario hablar de él desde un punto de vista más técnico para poder comprender mejor su funcionamiento.

Leela, como todos los motores de ajedrez, utiliza una función de evaluación para analizar y valorar las posiciones del árbol de juego. A diferencia de las funciones de evaluación más tradicionales, las cuales están compuestas por múltiples heurísticas hechas a mano por el programador, la función de evaluación de Leela usa una red neuronal entrenada sin ningún tipo de conocimiento humano (ya que es entrenada mediante “self-play” y “reinforcement learning”) que genera dos valores: la probabilidad de victoria asociada a la posición del árbol de juego que ha sido evaluada, la cual es denominada “value”, y un vector de probabilidad con los movimientos posibles desde dicha posición del árbol, conocido como “policy” [15].

Como también se ha dicho, además de una función de evaluación es necesario contar con una función de búsqueda para que la IA pueda tomar decisiones teniendo en cuenta jugadas futuras. El algoritmo que utiliza Leela para implementar esta función es conocido como PUCT (Predictor + Upper Confidence Bound tree search), el cual es muy similar a otro algoritmo utilizado para este propósito: Monte Carlo tree search [15]. Para guiar la búsqueda de este algoritmo, es decir, decidir que posiciones del árbol de juego merece la pena estudiar, se usa el valor de “policy” devuelto por la red neuronal, mientras que “value” es utilizado para evaluar dichas posiciones.

Por tanto, los movimientos que realiza la IA dependen directamente de “value” y de “policy”. Estos valores son las salidas, o cabezas de la red neuronal, y son generados a partir de la posición del árbol de juego que se pase como entrada a la red. Sin embargo, no es posible obtener los valores de “policy” y “value” a partir de una posición del árbol de juego de manera directa, sino que es necesario transformar esta posición utilizando diversas capas, filtros y funciones de activación.

Este conjunto de elementos forma lo que se conoce como el “cuerpo” de la red neuronal, el cual influye significativamente en los valores de las cabezas “policy” y “value”. Por este motivo, los desarrolladores de Leela ponen a disposición de los usuarios redes de diferentes topologías y tamaños, cada una diseñada para un hardware y un propósito determinado. Como norma general, cuanto mayor sea el tamaño de la red mayor es la calidad de sus predicciones, pero también el tiempo que necesita para calcularlas [16] [17].

3.3. MAIA CHESS

En el capítulo “2. Estudio de la situación actual” ya se ha introducido este proyecto. Sin embargo, puesto que todas las redes neuronales que se utilizarán en este trabajo se entrenarán mediante la “training pipeline” de Maia, es necesario estudiarlo con más detalle.

Como ya se ha dicho, uno de los objetivos de Maia es crear una inteligencia artificial de ajedrez cuyo comportamiento sea lo más humano posible. Sin embargo, debido a que la finalidad de Maia no es la búsqueda en este trabajo, no comparte varias de las condiciones fundamentales tratadas en este documento. No obstante, Maia utiliza una modificación de la “training pipeline” de Leela para crear y entrenar su red neuronal con partidas de jugadores reales, la cual utiliza posteriormente junto con el motor de Leela para evaluar las posiciones de la partida y efectuar los movimientos. Por tanto, este programa resulta de gran interés para el desarrollo de este trabajo.

En la actualidad existen dos variantes de Maia, las cuales comparten el objetivo de conseguir que una IA realice movimientos más humanos. La versión original de Maia, conocida como

Maia chess, logra este objetivo entrenando varias redes neuronales, cada una de las cuales con las partidas de los jugadores de un determinado rango de habilidad (ELO). De esta forma, esta versión está compuesta por 9 redes distintas, desde Maia-1100 (entrenada con las partidas de los jugadores con un ELO comprendido entre 1100 y 1199) hasta Maia-1900 (entrenada con las partidas de los jugadores con un ELO comprendido entre 1900 y 1999), cada una de ellas entrenada con 120000 partidas [12].

Gracias a esto, Maia chess consigue crear redes con un comportamiento más humano que el resto de IAs de ajedrez, sin embargo, lo hace de manera general para unos determinados grupos de habilidad, por lo que no es capaz de capturar las características propias del estilo de juego de un jugador en concreto. Por este motivo los creadores de Maia chess desarrollaron Maia individual, la cual entrena una red utilizando únicamente las partidas de un jugador determinado, consiguiendo imitar mucho mejor su estilo de juego de lo que lo haría su versión anterior. Además, a diferencia de Maia chess esta versión no entrena la red neuronal desde cero, sino que, independientemente del ELO del jugador que se quiera imitar, siempre parte de la red entrenada Maia-1900. Gracias a esto, no necesita tantas partidas como Maia chess para entrenar una red, sino que ya empieza a conseguir buenos resultados a partir de las 5000 [13].

Como se puede observar, entre las dos versiones de Maia, Maia individual es más parecida a la IA que se pretende construir en este trabajo. Por esta razón se usará su “training pipeline” para llevar a cabo el entrenamiento. Además, de la misma manera que se puede utilizar la red Maia-1900 como punto de partidas para el entrenamiento, también se pueden usar las redes antiguas de los jugadores. De esta forma, este programa permite la actualización de dichas redes neuronales, cumpliendo así uno de los objetivos de este trabajo. Por otra parte, también es importante mencionar que los desarrolladores de Maia individual ofrecen una serie de scripts para el procesamiento de las partidas, los cuales pueden ser utilizados para este trabajo si se les realizan algunas modificaciones.

Por último, en cuanto a la arquitectura de la red ambas versiones de Maia utilizan una red neuronal pequeña, de unos 1,2MB, ya que es ligera, rápida y puede ejecutarse en equipos

que tan solo disponen de una CPU. En la siguiente figura puede observarse la arquitectura de dicha red.

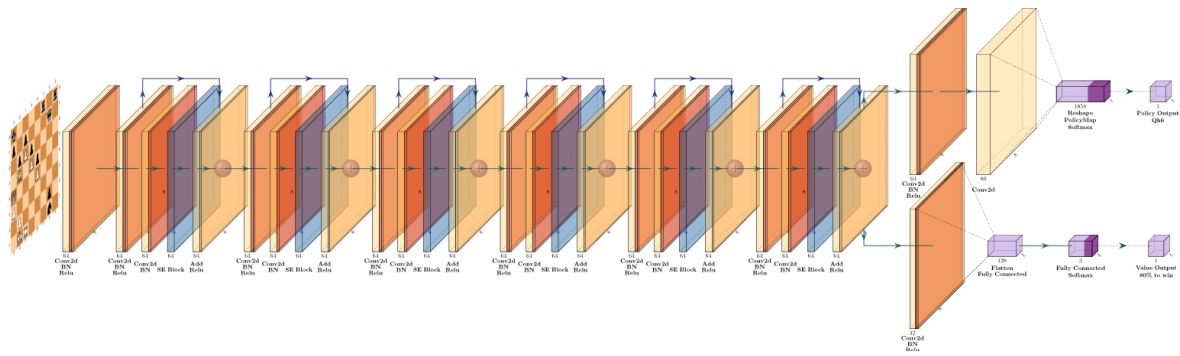


Figura 3.2 - Arquitectura de la red neuronal utilizada por Maia

Aunque en la figura anterior se muestre la cabeza “value”, tanto Maia chess como Maia individual la ignoran a la hora de decidir qué movimiento realizar. Esto se debe a que ninguna de las dos hace uso de la función de búsqueda, por lo que solo exploran un nivel del árbol de juego. Esta decisión es justificada por los desarrolladores de Maia al comparar una versión de Maia chess que no utiliza la función de búsqueda con otra versión que usa esta función para explorar 10 niveles del árbol de juego, ya que vieron que la versión que no emplea la función de búsqueda predice mejor los movimientos de los jugadores [12]. Sin embargo, esta comparación parece bastante arbitraria, puesto que la inmensa mayoría de los jugadores no analizaría diez niveles del árbol de juego para decir sus movimientos. De todas formas, esta no es una decisión que realmente le pertenezca al desarrollador, ya que es el jugador quien selecciona la profundidad máxima de exploración de la IA en el motor de ajedrez. Por estos motivos, la cabeza “value” sí se tendrá en cuenta en el desarrollo de este trabajo.

3.4. PGN-EXTRACT

PGN-extract es una herramienta de línea de comandos creada con el objetivo de facilitar la extracción y el formateo de archivos PGN [18]. En el caso de este trabajo, se utilizará para eliminar la información no necesaria de los archivos que contienen las partidas, como comentarios, evaluaciones de los movimientos, algunas etiquetas, etc.

3.5. TRAININGDATA-TOOL

Trainingdata-tool es una herramienta de línea de comandos desarrollada por los creadores de Leela. Esta permite transformar las partidas en formato PGN en los datos que se utilizan como entrada para entrenar la red neuronal [19].

3.6. BANKSIA GUI

Banksia GUI es una interfaz gráfica de ajedrez libre [20], la cual será utilizada para crear los libros de aperturas de los jugadores y para jugar contra las IAs desarrolladas a lo largo de este trabajo.

4. Trabajo realizado y resultados obtenidos

A lo largo de este capítulo se creará una inteligencia artificial de ajedrez que imite el estilo de juego de un jugador determinado y que cumpla con los tres objetivos propuestos en este trabajo. Estos son:

- Funcionamiento con muy pocos datos.
- Actualización con nuevas partidas, lo cual ya permite Maia.
- Tiempo de aprendizaje muy reducido.

Hay que tener en cuenta que esta IA ha de poder imitar el comportamiento de jugadores con diversos estilos de juego y diferentes niveles de habilidad, por lo que no sería suficiente con entrenar solo una. Por esta razón se entrenará un conjunto de 36 IAs diferentes utilizando una variedad de configuraciones y métodos distintos, pudiendo así observar y comparar los resultados obtenidos del entrenamiento y determinar de esta forma las mejores opciones para cumplir los objetivos propuestos.

4.1. EQUIPO DE PRUEBAS

Uno de los objetivos de este trabajo es conseguir que el tiempo de aprendizaje de la red neuronal sea lo más reducido posible. Este tiempo depende de varios factores, algunos de los cuales serán tratados a lo largo de este documento. Sin embargo, también depende del hardware del equipo en el que se ejecute el entrenamiento de la red. Por este motivo es necesario mostrar el hardware que conforma el equipo de pruebas.

En la siguiente tabla se muestran los componentes hardware del equipo de pruebas:

CPU	AMD Ryzen 7 1800x 4.0GHz
GPU	NVIDIA GeForce GTX 1080 8GB
RAM	DDR4 3200MHz 16GB CL15
HDD	1TB 7200RPM

Tabla 4.1 - Componentes hardware del equipo de pruebas

4.2. OBTENCIÓN Y PROCESAMIENTO DE DATOS

Como ya se sabe, para conseguir que la red neuronal de Leela imite el estilo de juego de un jugador determinado, esta ha de ser entrenada con una gran cantidad de partidas de dicho jugador. Por suerte, la disponibilidad de datos de partidas no es un problema en el campo del ajedrez, ya que existen servidores de ajedrez en línea que recopilan esta información y la ponen a disposición del público general. Por ejemplo, los desarrolladores de Maia utilizan Lichess [12] [13], una plataforma libre y de código abierto donde jugadores de todo el mundo juegan partidas de ajedrez a través de Internet. Lichess recopila la información de todas las partidas jugadas en sus servidores y las publica de manera mensual en su base de datos en formato PGN, permitiendo de esta forma la descarga de todas las partidas jugadas desde enero de 2013 hasta la actualidad.

Debido a la inmensa cantidad de datos que ofrece Lichess, se ha optado por usar su base de datos para obtener las partidas que se utilizarán durante el entrenamiento de las IAs. Además, al ser también empleada por Maia, se pueden aprovechar los scripts proporcionados por esta para procesar los archivos PGN descargados sin tener que realizar demasiadas modificaciones.

Una vez decidida la fuente de los datos, se debe determinar qué partidas se utilizarán para el entrenamiento. Para empezar, se descargaron todas las partidas jugadas en los servidores de Lichess desde enero de 2014 hasta diciembre de 2016. Esta selección permite dos cosas: asegurarse de que hay un número suficiente de jugadores con una gran cantidad de partidas, y que las partidas hayan sido jugadas a lo largo de un periodo de tiempo lo suficientemente largo, de forma que se pueda observar el cambio de estilo de cada jugador a medida que este va jugando. Por otro lado, la razón por la que se decide descargar partidas antiguas en vez

de más recientes es porque la base de jugadores de Lichess ha aumentado enormemente en los últimos años. Por ejemplo, el archivo PGN comprimido que contiene todas las partidas jugadas en enero de 2024 pesa 32,4GB, mientras que el archivo que contiene las partidas de enero de 2014 tan solo pesa 111MB. Teniendo en cuenta que el equipo utilizado solo dispone de 1TB de almacenamiento, resultaría imposible descargar partidas recientes jugadas en un periodo de tiempo tan largo.

Como ya se ha dicho, cada uno de estos archivos PGN contiene todas las partidas jugadas durante un mes determinado, las cuales siguen un formato similar al que se muestra en la siguiente figura.

```
[Event "Rated Blitz game"]
[Site "https://lichess.org/JGxEntwR"]
[White "Cazado"]
[Black "coralines"]
[Result "0-1"]
[UTCDate "2014.07.01"]
[UTCTime "16:35:25"]
[WhiteElo "1697"]
[BlackElo "1761"]
[WhiteRatingDiff "-10"]
[BlackRatingDiff "+9"]
[ECO "D05"]
[Opening "Colle System #2"]
[TimeControl "180+0"]
[Termination "Normal"]

1. d4 { [%eval 0.2] } 1... d5 { [%eval 0.31] } 2. Nf3 { [%eval 0.2] } 2... Nf6 { [%eval 0.2] } 3. e3 { [%eval 0.19] } 3... e6 { [%eval 0.12] } 4. Bd3 { [%eval 0.04] } 4... Be7 { [%eval 0.08] } 5. O-O { [%eval 0.06] } 5... O-O { [%eval 0.08] } 6. c4 { [%eval 0.08] } 6... dxc4 { [%eval 0.12] } 7. Bxc4 { [%eval 0.12] } 7... c6 { [%eval 0.3] } 8. Nc3 { [%eval 0.48] } 8... b5 { [%eval 0.2] } 9. Be2 { [%eval 0.17] } 9... a6 { [%eval 0.18] } 10. b3 { [%eval 0.0] } 10... Nd5 { [%eval 0.38] } 11. Bb2 { [%eval 0.41] } 11... Bb7 { [%eval 0.39] } 12. Rc1 { [%eval 0.34] } 12... Nd7 { [%eval 0.27] } 13. Ne4 { [%eval 0.3] } 13... f5 { [%eval 0.3] } 14. Nc3 { [%eval 0.26] } 14... Bb4 { [%eval 0.42] } 15. Nxd5 { [%eval 0.31] } 15... exd5 { [%eval 0.43] } 16. Qc2 { [%eval 0.19] } 16... Qe7 { [%eval 0.18] } 17. Bd3 { [%eval 0.15] } 17... g6 { [%eval 0.23] } 18. a3?! { [%eval -0.31] } 18... Bxa3 { [%eval -0.23] } 19. Bxa3 { [%eval -0.52] } 19... Qxa3 { [%eval -0.34] } 20. Ng5 { [%eval -0.5] } 20... Qe7 { [%eval -0.68] } 21. f4?! { [%eval -1.61] } 21... Qxe3+ { [%eval -1.6] } 22. Kh1 { [%eval -1.4] } 22... Qxd4 { [%eval -1.69] } 23. Ne6 { [%eval -1.53] } 23... Qf6 { [%eval -1.53] } 24. Nxf8 { [%eval -1.49] } 24... Rxf8 { [%eval -1.41] } 25. g3?! { [%eval -2.37] } 25... c5 { [%eval -2.2] } 26. Kg2? { [%eval -4.76] } 26... c4? { [%eval -3.46] } 27. bxc4? { [%eval -6.32] } 27... dxc4+ { [%eval -6.27] } 28. Kg1? { [%eval -7.99] } 28... cxd3? { [%eval -6.4] } 29. Qxd3?! { [%eval -7.25] } 29... Qb6+ { [%eval -7.03] } 30. Rf2 { [%eval -7.17] } 30... Nf6 { [%eval -7.01] } 31. Qb3+ { [%eval -7.37] } 31... Bd5 { [%eval -7.41] } 32. Qb4 { [%eval -7.83] } 32... Ne4 { [%eval -7.95] } 33. Qd4? { [%eval #-5] } 33... Qxd4 { [%eval #-4] } 0-1
```

Figura 4.1 - Ejemplo de partida descargada de la base de datos de Lichess

Como se puede ver, la partida mostrada contiene mucha más información que la partida de la figura 3.1, la cual tendrá que eliminarse antes de poder ser utilizada para el entrenamiento. Sin embargo, resultan de gran interés las etiquetas “WhiteElo” (nivel de habilidad del jugador que juega con blancas), “BlackElo” (nivel de habilidad del jugador que juega con

negras), y “UTCTime” (hora de comienzo de la partida), las cuales serán de gran utilidad más adelante.

Tras descargar las partidas, las cuales se encuentran distribuidas en 36 archivos PGN, estas serán procesadas con una serie de scripts, los cuales son modificaciones de los proporcionados por Maia. Al procesarlas con dichos scripts se busca crear un primer conjunto de datos similar al utilizado en Maia individual, de forma que se pueda crear un grupo inicial de IAs entrenadas con dichas partidas que sirva como referencia para las diferentes pruebas que se lleven a cabo.

El primero de estos scripts se encarga de extraer de los archivos PGNs información básica de los jugadores, la cual es almacenada en un fichero csv de forma que pueda ser utilizada por el resto de scripts. El archivo csv generado por este script está formado por dos columnas, donde la primera corresponde al nombre del jugador y la segunda al número de partidas que ha jugado dicha persona. Es importante mencionar que este archivo tiene una peculiaridad, y es que para seguir el criterio utilizado en Maia individual para la selección de jugadores, el fichero solo contiene a aquellos que hayan disputado más de 5000 partidas [13].

Una vez creado el archivo csv ya se puede ejecutar el siguiente script, el cual se encarga de dos tareas. Por un lado, agrupa por jugador las partidas que aparecen en el fichero csv, de forma que cada uno de ellos disponga de su propio archivo PGN con todas sus partidas. Por otra parte, el script también extrae varias estadísticas relevantes de estos jugadores, más específicamente el porcentaje de victorias, el porcentaje de partidas jugadas con blancas, el ELO medio y el número de partidas jugadas por mes.

Tras ejecutar los scripts anteriores, quedan un total de 639 jugadores que pueden utilizarse para realizar las pruebas. Sin embargo, debido a las limitaciones del hardware no sería viable realizar pruebas con un conjunto de jugadores de este tamaño, por lo que es necesario realizar una selección manual. Con el objetivo de crear un conjunto pequeño, pero lo suficientemente grande para que las pruebas tengan validez, se han seleccionado 36 jugadores siguiendo los siguientes criterios:

- **Ha de pertenecer a alguno de los rangos de ELO determinados por Maia:** tanto en Maia chess como en Maia individual se utilizan 9 rangos de ELO, desde el rango 1100 (el cual contiene a todos los jugadores con un ELO entre 1100 y 1199) hasta el rango 1900 (el cual recoge a todos los jugadores con un ELO entre 1900 y 1999) [12] [13]. Con el objetivo de que cada uno de estos rangos sea representado adecuadamente durante las pruebas, se intenta seleccionar a 4 jugadores por rango.
- **El porcentaje de victorias ha de ser equilibrado:** el porcentaje de victorias de cada jugador no debe ser ni muy alto ni muy bajo. Un porcentaje excesivamente alto de victorias puede ser señal de que el jugador esté haciendo trampas, y uno muy bajo puede ser indicio de que el jugador esté dejándose ganar voluntariamente.
- **El porcentaje de partidas jugadas con blancas ha de ser equilibrado:** al igual que con el porcentaje de victorias, el porcentaje de partidas que cada jugador juega como blancas no debe ser ni muy alto ni muy bajo. Un porcentaje demasiado alto puede indicar que el jugador se aproveche de la ventaja estratégica que supone jugar con este bando para aumentar su ELO, y uno excesivamente bajo puede significar que el jugador esté haciendo uso de un motor de ajedrez para ganar.

Teniendo en cuenta los criterios anteriores se han seleccionado 36 jugadores, con los cuales se llevarán a cabo todas las pruebas realizadas a lo largo de este trabajo. Sin embargo, sus partidas aún siguen en formato PGN, por lo que antes de poder entrenar las redes neuronales se han de convertir en el formato de que utiliza Leela para el entrenamiento.

Para realizar esta conversión, se hace uso de uno de los scripts proporcionados por Maia. Al igual que los otros scripts, este realiza varias tareas. Primero se encarga de dividir las partidas de cada jugador en los siguientes conjuntos:

- **Entrenamiento:** este conjunto es utilizado para modificar los pesos de la red. Es usado por la red neuronal durante el entrenamiento para extraer el estilo del jugador.
- **Test:** este conjunto es utilizado para comprobar que no se produzca un sobreajuste de la red. Es decir, se usa para comprobar que la red es capaz de predecir movimientos que no haya visto durante el entrenamiento.

Una vez divididas las partidas en los conjuntos anteriores, el script utiliza la herramienta “PGN-extract” para eliminar la información innecesaria de las partidas. Más concretamente, el script elimina todos los comentarios, evaluaciones de jugadas, y las etiquetas que no pertenezcan al STR. Por último, este utiliza la herramienta “trainingdata-tool” para convertir las partidas en formato PGN en el formato utilizado por Leela para entrenar las redes neuronales.

Tras ejecutar este script, ya se dispone de los datos necesarios para realizar el entrenamiento de las 36 IAs por medio de la “training pipeline” proporcionada por Maia.

4.3. BASE

En este apartado se llevará a cabo el entrenamiento del conjunto de IAs que servirá como base para medir los resultados de las pruebas. La creación de este conjunto base se ha explicado de manera detallada en el apartado anterior; está formado por 36 jugadores, los cuales tienen más de 5000 partidas, se encuentran en el rango de ELO utilizado por Maia, y tienen unos porcentajes equilibrados de victorias y de partidas jugadas con blancas. En la siguiente tabla se pueden ver dichas características de los jugadores:

Jugador	Partidas	ELO medio	Victorias (%)	Partidas con blancas (%)
1	14632	1139	43,719	1,551
2	8686	1190	30,059	59,072
3	15589	1196	38,693	45,609
4	14140	1203	21,060	39,009
5	23981	1276	29,773	48,917
6	21417	1236	51,879	49,843
7	14324	1310	35,534	46,160
8	11466	1298	58,939	44,662
9	41667	1423	31,838	49,900
10	18255	1360	47,795	46,168
11	17907	1386	29,078	39,219
12	26098	1352	40,631	48,777
13	23679	1466	22,433	44,157
14	40089	1499	35,491	59,510
15	19710	1454	43,155	50,360
16	13458	1467	42,628	47,295

17	25496	1585	36,499	50,380
18	18197	1576	43,551	50,700
19	18897	1573	42,287	47,510
20	26257	1517	42,251	49,735
21	77562	1648	51,876	49,680
22	41066	1659	46,712	49,196
23	26992	1681	37,366	48,525
24	32933	1635	37,400	48,392
25	19075	1738	50,390	47,816
26	39861	1762	47,728	53,448
27	53296	1746	43,567	49,883
28	24569	1764	37,433	51,829
29	23884	1862	52,009	56,602
30	31283	1884	54,933	48,956
31	33110	1847	63,394	49,791
32	27525	1865	43,945	49,416
33	31184	2004	74,634	60,271
34	39986	1976	53,353	48,692
35	23623	1964	70,799	49,379
36	50017	1987	66,283	49,133

Tabla 4.2 - Información de los jugadores del conjunto base

Viendo la tabla se puede observar que existen algunas excepciones respecto a las características mencionadas anteriormente. En primer lugar, El jugador 1 tiene un porcentaje de partidas disputadas con blancas extremadamente bajo (1,551%). Esto puede significar que esté jugando con negras deliberadamente para poder hacer uso de un motor de ajedrez y ganar así una gran cantidad de partidas. Sin embargo, esta hipótesis no parece probable por los siguientes motivos:

- **Su porcentaje de victorias es normal.** A pesar de que su porcentaje de partidas jugadas con blancas es muy bajo, su porcentaje de victorias es equilibrado. Si hiciera uso de un motor de ajedrez este valor sería mucho más elevado.
- **Tiene un ELO bajo.** Esto significa que el jugador no es capaz de derrotar de manera regular a contrincantes con un ELO similar o mayor que el suyo. Al igual que en el punto anterior, si hiciera uso de un motor de ajedrez este valor sería mayor.

La razón por la que se utiliza a este jugador para realizar las pruebas se debe a que no existen más jugadores que se encuentren en el rango de ELO 1100 y que tengan más de 5000 partidas

a sus espaldas. Debido a esto, y a que parece que no está haciendo trampas, se opta por incluir a este jugador en el conjunto.

Otra excepción es el jugador 33, el cual no pertenece al rango de ELO utilizado por Maia ([1100, 1999]). No obstante, similar a lo que ocurre con el jugador 1, no existen más jugadores en el rango de ELO 1900. Debido a que no está tan alejado del rango objetivo, este jugador también se añade al conjunto.

Es importante mencionar que no todas las partidas que aparecen en la tabla se utilizarán para crear el conjunto de entrenamiento. Siguiendo la configuración por defecto de Maia individual, se utilizará el 80% de las partidas para crear este conjunto, mientras que el 20% restante se empleará para constituir el conjunto de test.

En este punto ya se podría comenzar el entrenamiento del conjunto base, sin embargo, antes de empezar es necesario comentar las métricas que se van a obtener de este proceso.

Estas métricas se dividen en dos tipos: las que se obtienen a partir del conjunto de entrenamiento y las que se obtienen a partir del conjunto de test [15]. Del conjunto de entrenamiento de cada jugador se obtienen los siguientes datos:

- **Step:** paso de entrenamiento. Cada paso supone el procesamiento de un lote de datos y una actualización del gradiente.
- **Policy loss:** es un número que indica cuanto difiere el movimiento que predice la red del que realiza el jugador. Además, es uno de los términos que la red neuronal intenta minimizar.
- **Value loss:** es un número que indica cuánto difiere la probabilidad de victoria que pronostica la red con la probabilidad real. Además, es uno de los términos que la red neuronal intenta minimizar.
- **Total:** es el error total de la red.

Por otra parte, del conjunto de test de cada jugador se obtienen las siguientes métricas:

- **Step:** igual que para el conjunto de entrenamiento.
- **Policy accuracy:** porcentaje de precisión de la cabeza “policy”.

- **Value accuracy:** porcentaje de precisión de la cabeza “value”.

Es importante mencionar que, a pesar de que la “training pipeline” calcula y utiliza todos los datos anteriores, es necesario modificarla para que además almacene toda esta información formateada en ficheros csv, de forma que pueda ser analizada posteriormente.

Una vez tratadas las métricas, ya se puede comenzar con el entrenamiento del conjunto. Con el objetivo de que este proceso sea lo más parecido posible al utilizado para entrenar las IAs de Maia individual, se utilizó la misma configuración de hiperparámetros que en dicho estudio. Esto significa que se han empleado 30000 pasos de entrenamiento y un learning rate variable que sigue el siguiente esquema [13]:

Pasos de entrenamiento	Learning rate
[0, 15000]	0,01
(15000, 20000]	0,001
(20000, 25000]	0,0001
(25000, 30000]	0,00001

Tabla 4.3 - Configuración por defecto del learning rate

Una vez completada la ejecución del proceso de entrenamiento, se ha hecho uso de las métricas anteriores para crear un conjunto de gráficas con las que poder analizar los resultados obtenidos. Sin embargo, debido a su elevado número, estas gráficas aparecen recogidas en el anexo.

Aparte de las gráficas, tras el entrenamiento se puede ver que, utilizando 30000 pasos, este proceso supone una hora y media de tiempo de ejecución por cada uno de los 36 jugadores que forman el conjunto. A pesar de que este tiempo es muy elevado para los objetivos que se persiguen en este trabajo, si se observan las gráficas de porcentaje de precisión de “policy” y “value” de todos los jugadores, se puede observar que en la gran mayoría de los casos la mejora más significativa en la precisión se produce durante los primeros 1000 pasos. Esto es una buena noticia, ya que el tiempo de ejecución de 1000 pasos de entrenamiento es de unos 3 minutos, lo que permitiría cumplir con el objetivo de conseguir un tiempo de ejecución reducido.

Sin embargo, este conjunto base tiene un problema, ya que se ha creado utilizando el programa empleado por los desarrolladores de Maia para separar las partidas en los conjuntos de entrenamiento y test. Este programa no tiene ningún criterio para seleccionar las partidas, simplemente lee el archivo en el que se encuentran todas las partidas de un jugador y las va añadiendo a los conjuntos de entrenamiento y test hasta alcanzar los porcentajes que previamente se le hayan indicado. Esto supone un problema, ya que las partidas del archivo no están ordenadas cronológicamente, aunque a primera vista lo parezca, lo que hace que la selección del conjunto de test sea inconsistente.

Al analizar las gráficas del anexo que recogen la distribución temporal de las partidas de los conjuntos de test y de entrenamiento de todos los jugadores, se puede observar este fenómeno. Por ejemplo, la siguiente gráfica pertenece al jugador 36.

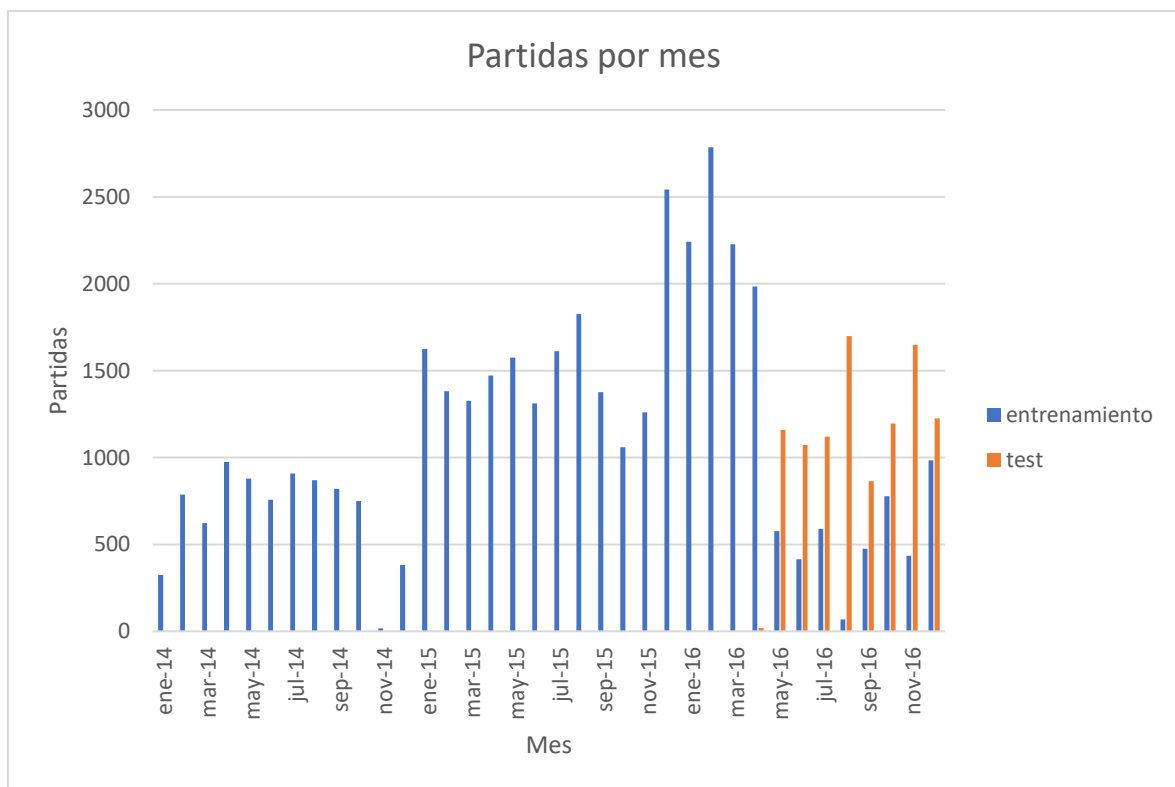


Figura 4.2 - Número de partidas por mes del jugador 36

Como se puede observar, las partidas del conjunto de test se concentran durante los últimos meses del estudio, pero esto no significa que no haya también partidas disputadas durante dichos meses en el conjunto de entrenamiento.

En cambio, no se puede decir lo mismo para el jugador 29.

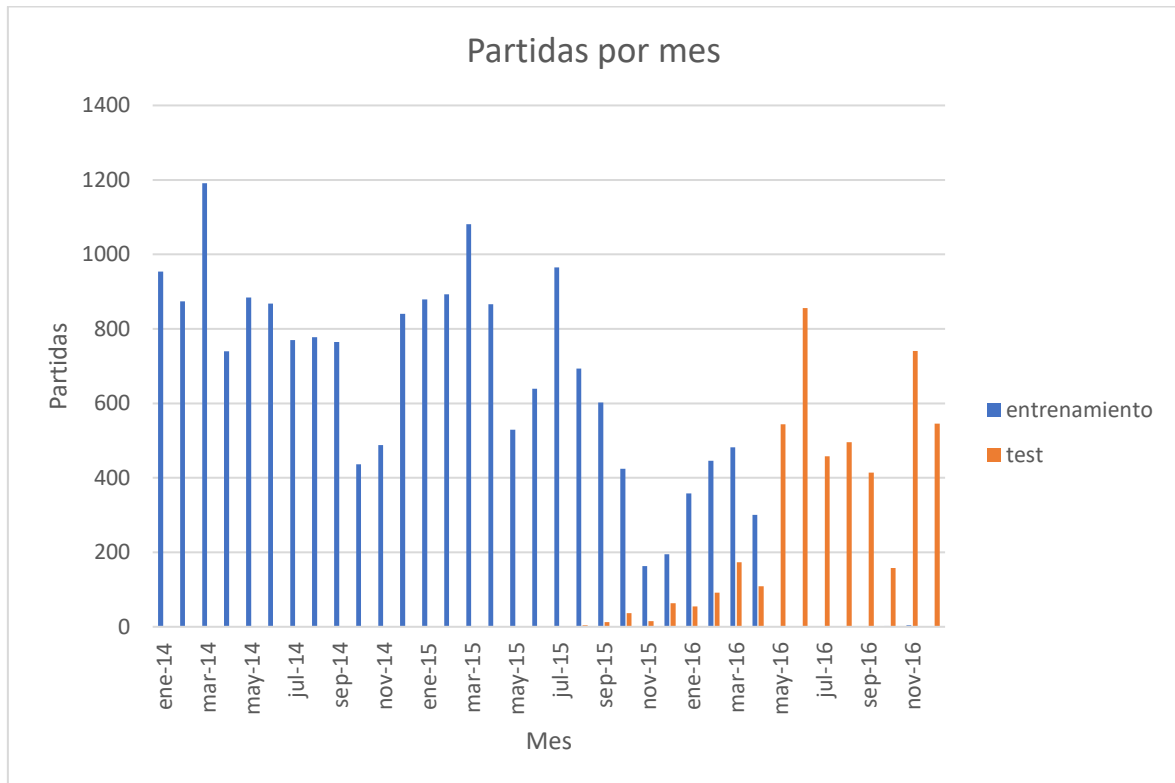


Figura 4.3 - Número de partidas por mes del jugador 29

Se puede ver que, a pesar de que las partidas del conjunto de test siguen concentrándose durante los meses finales, en este caso el programa apenas ha utilizado partidas de los últimos 8 meses para crear el conjunto de entrenamiento.

Sin embargo, el caso más claro de esta inconsistencia aparece con el jugador 21.

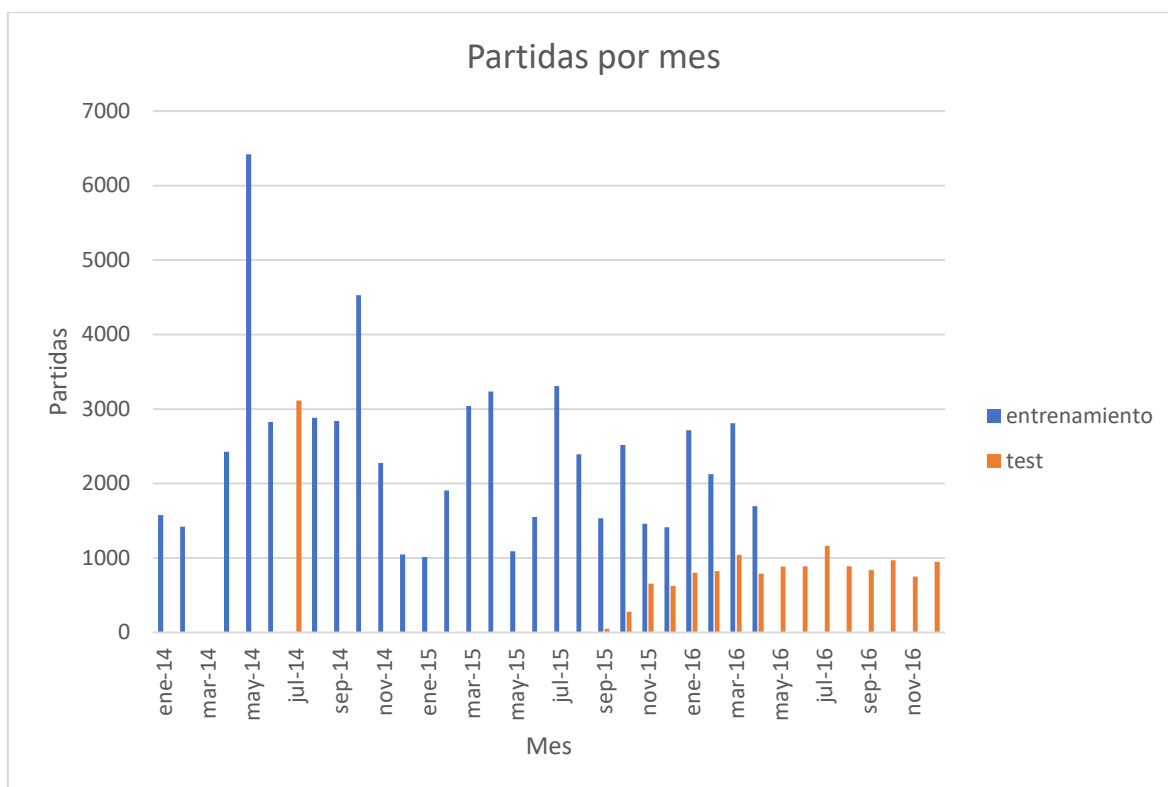


Figura 4.4 - Número de partidas por mes del jugador 21

En este caso, en vez de emplear únicamente las partidas de los últimos meses, también se utilizan todas las partidas disputadas a lo largo del mes de julio de 2014.

De todas formas, y más importante aún, para los objetivos de este trabajo no tiene sentido comprobar la calidad de las predicciones de las redes neuronales con partidas más antiguas que las que forman el conjunto de entrenamiento. Lo que se quiere conseguir con este trabajo es una IA que juegue con el mismo estilo que tiene el jugador en el momento en el que esta se crea, es decir, que juegue de la misma manera que el jugador lo haría en las siguientes partidas. Por tanto, el conjunto de test no debe estar formado por partidas elegidas de forma aleatoria, sino que debe estar formado por las partidas que el jugador disputó inmediatamente después de la partida más reciente del conjunto de entrenamiento. Además, este conjunto de test tampoco debe ser muy grande, ya que se corre el riesgo de que el jugador utilice más de un estilo en las partidas que lo componen. Por estos motivos, se volverá a crear el conjunto base utilizando conjuntos de test compuestos por las últimas 100 partidas que hayan disputado los jugadores.

Obtener estas partidas no es tan sencillo como pueda parecer, ya que estas no están ordenadas cronológicamente y los archivos que las contienen están formados por cientos de miles de líneas. Por esta razón fue necesario crear dos scripts. El primero de estos, utilizando la información de las etiquetas “UTCDate” y “UTCTime” obtenidas de las partidas, y el número de partidas totales y mensuales de los jugadores, identifica las 100 partidas más recientes y las escribe en un fichero independiente. En cambio, el segundo script, valiéndose de la misma información que utiliza el primero, también se encarga de identificar estas 100 partidas, pero para guardar las restantes en un fichero aparte. De esta forma, el primer script crea el conjunto de test, mientras que el segundo crea el de entrenamiento.

Aparte de los cambios en los conjuntos de entrenamiento y test, otra diferencia con el entrenamiento del conjunto base anterior será el número de pasos empleado. Como ya se ha dicho, utilizar 30000 pasos de entrenamiento es incompatible con los objetivos propuestos en este trabajo, siendo mucho más viable el uso de 1000. Por tanto, con la intención de reducir el tiempo necesario para entrenar el conjunto, pero sin sacrificar la posible información relevante que se pueda obtener en pasos posteriores a 1000, el entrenamiento del nuevo conjunto base se llevará a cabo empleando 10000 pasos.

Tras obtener las métricas del entrenamiento y crear las subsecuentes gráficas, las cuales se pueden encontrar en el anexo, lo primero que se puede observar es que no se ha podido entrenar al jugador 1. Esto se debe a que sus últimas 100 partidas, es decir, las partidas que componen su conjunto de test, las ha jugado como negras. Debido a esto el programa no permite su entrenamiento y, por tanto, no existen métricas de este jugador en este conjunto.

Por otra parte, al observar las gráficas de precisión de “policy” y “value” se puede observar que, al igual que sucedía con el conjunto base anterior, la mejora más significativa ocurre durante los primeros 1000 pasos de entrenamiento. Además, si se comparan estas nuevas precisiones con las obtenidas anteriormente se puede ver que, por lo general, son bastante similares.

Con el objetivo de visualizar mejor los cambios en la precisión según aumentan los pasos de entrenamiento, se ha creado una tabla que compara la precisión inicial, es decir, la precisión

de la red Maia-1900, con la precisión obtenida a los 100, 500 y 1000 pasos de entrenamiento. Como ya se ha dicho, para cumplir los objetivos de este trabajo sería viable utilizar 1000 pasos, los cuales equivalen a unos 3 minutos de ejecución. Sin embargo, merece la pena estudiar si se consiguen buenas precisiones utilizando menos, ya que el tiempo de ejecución de 500 pasos es de un minuto y medio, y el de 100 es de tan solo 18 segundos.

Esta tabla está compuesta por varias columnas, las cuales significan lo siguiente:

- **Jugador:** el jugador cuya precisión se está analizando.
- **P (x):** precisión de la cabeza “policy” a los x pasos de entrenamiento. En el caso de P (0), esta indica la precisión de “policy” de Maia-1900.
- **V (x):** precisión de la cabeza “value” a los x pasos de entrenamiento. En el caso de V (0), esta indica la precisión de “value” de Maia-1900.
- **MP (x):** mejora de la precisión de la cabeza “policy” a los x pasos de entrenamiento comparada con la precisión de Maia-1900. Las celdas de esta columna aparecen coloreadas según su valor, de forma que si la precisión en el paso x es superior a la obtenida por Maia-1900 la casilla aparece coloreada de verde, si es inferior coloreada de rojo y en caso de ser iguales coloreada de amarillo.
- **MV (x):** mejora de la precisión de la cabeza “value” a los x pasos de entrenamiento comparada con la precisión de Maia-1900. Las celdas de esta columna aparecen coloreadas según su valor, de forma que si la precisión en el paso x es superior a la obtenida por Maia-1900 la casilla aparece coloreada de verde, si es inferior coloreada de rojo y en caso de ser iguales coloreada de amarillo.

A continuación se encuentra dicha tabla.

Jugador	P (0)	V (0)	P (100)	V (100)	MP (100)	MV (100)	P (500)	V (500)	MP (500)	MV (500)	P (1000)	V (1000)	MP (1000)	MV (1000)
1	-	-	-	-	-	-	-	-	-	-	-	-	-	-
2	41,0156	63,6719	40,625	77,3438	-0,3906	13,6719	47,6562	81,6406	6,6406	17,9687	47,2656	79,2969	6,25	15,625
3	50,7812	51,5625	51,9531	62,8906	1,1719	11,3281	58,5938	64,0625	7,8126	12,5	48,8281	66,4062	-1,9531	14,8437
4	45,7031	69,1406	44,9219	80,8594	-0,7812	11,7188	46,0938	75,3906	0,3907	6,25	46,4844	84,7656	0,7813	15,625
5	47,6562	60,9375	47,2656	73,4375	-0,3906	12,5	44,5312	65,625	-3,125	4,6875	51,9531	73,4375	4,2969	12,5
6	40,2344	57,0312	49,2188	65,625	8,9844	8,5938	48,4375	56,6406	8,2031	-0,3906	50,7812	72,6562	10,5468	15,625
7	45,3125	58,5938	54,2969	59,375	8,9844	0,7812	53,5156	57,0312	8,2031	-1,5626	56,25	57,8125	10,9375	-0,7813
8	46,4844	80,0781	64,8438	75	18,3594	-5,0781	58,9844	76,5625	12,5	-3,5156	58,9844	71,875	12,5	-8,2031
9	45,7031	55,4688	60,9375	71,4844	15,2344	16,0156	59,375	74,6094	13,6719	19,1406	62,5	79,2969	16,7969	23,8281
10	45,7031	57,4219	46,4844	56,25	0,7813	-1,1719	48,0469	65,2344	2,3438	7,8125	55,8594	61,3281	10,1563	3,9062
11	45,7031	61,7188	47,6562	59,375	1,9531	-2,3438	51,5625	74,2188	5,8594	12,5	53,9062	77,7344	8,2031	16,0156
12	56,25	67,9688	52,3438	61,3281	-3,9062	-6,6407	54,6875	59,375	-1,5625	-8,5938	52,3438	58,2031	-3,9062	-9,7657
13	50,7812	58,2031	53,9062	67,5781	3,125	9,375	47,6562	81,25	-3,125	23,0469	52,3438	80,0781	1,5626	21,875
14	41,4062	44,1406	55,8594	69,5312	14,4532	25,3906	54,6875	62,8906	13,2813	18,75	55,4688	65,625	14,0626	21,4844
15	50	54,6875	60,5469	60,1562	10,5469	5,4687	55,4688	66,7969	5,4688	12,1094	52,3438	62,1094	2,3438	7,4219
16	53,9062	57,0312	46,875	70,3125	-7,0312	13,2813	56,25	64,0625	2,3438	7,0313	53,5156	66,0156	-0,3906	8,9844
17	46,0938	64,0625	50,3906	68,75	4,2968	4,6875	58,2031	63,2812	12,1093	-0,7813	58,9844	57,0312	12,8906	-7,0313
18	52,3438	68,75	52,3438	48,4375	0	-20,313	51,1719	48,8281	-1,1719	-19,9219	57,0312	46,0938	4,6874	-22,6562
19	57,4219	64,4531	58,2031	57,4219	0,7812	-7,0312	57,8125	62,8906	0,3906	-1,5625	51,9531	58,9844	-5,4688	-5,4687
20	51,9531	64,0625	50,3906	67,5781	-1,5625	3,5156	51,1719	64,8438	-0,7812	0,7813	52,3438	59,375	0,3907	-4,6875
21	42,1875	65,625	41,7969	61,7188	-0,3906	-3,9062	40,625	64,0625	-1,5625	-1,5625	42,5781	58,5938	0,3906	-7,0312
22	53,5156	80,0781	54,6875	72,6562	1,1719	-7,4219	55,8594	66,7969	2,3438	-13,2812	64,0625	65,625	10,5469	-14,4531
23	51,1719	63,6719	53,125	56,25	1,9531	-7,4219	51,9531	73,8281	0,7812	10,1562	59,375	62,1094	8,2031	-1,5625
24	48,0469	47,6562	47,6562	51,9531	-0,3907	4,2969	45,7031	57,0312	-2,3438	9,375	49,6094	56,6406	1,5625	8,9844
25	51,1719	52,7344	59,7656	57,4219	8,5937	4,6875	51,5625	59,375	0,3906	6,6406	51,9531	61,7188	0,7812	8,9844

26	41,4062	47,2656	41,4062	64,4531	0	17,1875	57,8125	55,0781	16,4063	7,8125	53,5156	58,9844	12,1094	11,7188
27	53,5156	70,7031	48,8281	57,4219	-4,6875	-13,281	48,0469	67,5781	-5,4687	-3,125	58,2031	58,2031	4,6875	-12,5
28	47,6562	58,5938	52,7344	62,1094	5,0782	3,5156	53,5156	61,7188	5,8594	3,125	50,7812	62,1094	3,125	3,5156
29	58,2031	58,5938	50,7812	62,8906	-7,4219	4,2968	54,6875	65,2344	-3,5156	6,6406	55,4688	55,8594	-2,7343	-2,7344
30	54,2969	55,4688	52,7344	55,4688	-1,5625	0	55,0781	61,3281	0,7812	5,8593	60,9375	57,4219	6,6406	1,9531
31	56,25	71,875	61,3281	77,3438	5,0781	5,4688	58,2031	78,125	1,9531	6,25	62,5	70,3125	6,25	-1,5625
32	48,8281	71,4844	57,0312	67,1875	8,2031	-4,2969	49,2188	67,9688	0,3907	-3,5156	60,1562	58,9844	11,3281	-12,5
33	63,6719	73,8281	53,125	81,25	-10,5469	7,4219	53,5156	70,7031	-10,1563	-3,125	58,9844	85,5469	-4,6875	11,7188
34	51,9531	51,9531	48,4375	50,7812	-3,5156	-1,1719	50,7812	50,7812	-1,1719	-1,1719	50,3906	51,1719	-1,5625	-0,7812
35	52,7344	71,875	58,9844	78,5156	6,25	6,6406	53,9062	72,6562	1,1718	0,7812	58,5938	67,1875	5,8594	-4,6875
36	47,6562	64,0625	51,9531	55,8594	4,2969	-8,2031	48,8281	50,7812	1,1719	-13,2813	46,4844	56,6406	-1,1718	-7,4219

Tabla 4.4 – Mejora de precisión (“policy” y “value”) del conjunto base respecto a Maia-1900 en los pasos de entrenamiento 100, 500 y 1000

Como se puede ver, no todos los jugadores consiguen mejorar las precisiones de “policy” y “value” alcanzadas por Maia-1900. Algunos tienen peores precisiones de “policy”, otros de “value”, y unos pocos de ambas. Esto es normal, ya que existen varios factores que afectan a estos resultados. Por ejemplo, los pasos que aparecen en la tabla son extremadamente reducidos, los conjuntos de test utilizados son pequeños y están formados por partidas más recientes que las de los conjuntos de entrenamiento. Además, también hay que tener en cuenta que no todos los jugadores son iguales, hay algunos que son más sencillos de predecir que otros.

También se puede observar que no siempre se consiguen valores más altos de precisión al aumentar el número de pasos. Esto puede verse en las gráficas de precisión de cada uno de los jugadores, donde estos valores sufren muchos altibajos a lo largo de la ejecución del programa. Esto puede deberse al learning rate utilizado. A pesar de que este es el mismo que han utilizado los desarrolladores de Maia individual, hay que tener en cuenta que este es un learning rate variable optimizado para el uso de 30000 pasos, por lo que puede ser demasiado alto si se utilizan muchos menos.

De todas formas, esta tabla ha sido creada utilizando la configuración por defecto de Maia individual, empleando para cada jugador un número de partidas muy elevado, al menos para los objetivos buscados en este trabajo. Por lo que esta tabla servirá de referencia para las siguientes pruebas.

4.4. REDUCCIÓN DE PARTIDAS

Uno de los objetivos principales de este trabajo es conseguir que la IA pueda funcionar al ser entrenada con muy pocas partidas. Por este motivo, a lo largo de este apartado se crearán 5 conjuntos de pruebas, cada uno de los cuales con menos partidas que el anterior, de forma que se pueda observar cómo afecta esta disminución de partidas a las precisiones de las cabezas “policy” y “value” de la red neuronal.

Según los desarrolladores de Maia individual, por debajo de 5000 partidas se pierde demasiada precisión en las predicciones de la red, llegando al punto de que utilizar 1000 resulta inviable [13]. Para poder comprobar estas afirmaciones, se entrenarán conjuntos de 1000, 500, 250, 100 y 50 partidas, y se compararán con los resultados obtenidos del entrenamiento del conjunto base.

Para poder utilizar los resultados del conjunto base como referencia, los nuevos conjuntos han de crearse utilizando a los mismos jugadores, lo que significa que es necesario descartar un gran número de partidas. El borrado de estas no puede realizarse de cualquier manera, para no afectar a los resultados deben ser eliminadas cronológicamente, de forma que no se produzcan saltos temporales entre las partidas restantes. Por suerte, para realizar esta tarea puede utilizarse el script creado en el apartado anterior para generar el conjunto de entrenamiento. De esta manera, las partidas se borrarán en orden desde las más recientes hasta las más antiguas, pudiendo simular así por medio de los diferentes conjuntos la evolución de los jugadores y de sus IAs con el paso del tiempo.

Al igual que con el conjunto base, en estos nuevos conjuntos se utilizarán 100 partidas para crear el conjunto de test, más concretamente las 100 más recientes. Por ejemplo, para crear el conjunto 1000 se utilizarán los 36 jugadores del conjunto base, pero utilizando 1100 partidas por cada uno de ellos, de forma que sus 100 partidas más recientes son utilizadas para crear sus conjuntos de test, mientras que las 1000 restantes se emplean para formar sus conjuntos de entrenamiento.

Las siguientes 5 tablas recogen las características de los jugadores de los conjuntos 1000, 500, 250, 100 y 50 respectivamente. Sin embargo, a diferencia de la tabla 4.2 del apartado anterior, la cual recoge la información obtenida de todas las partidas, estas tablas solo muestran los datos extraídos de las partidas que forman el conjunto de entrenamiento.

Jugador	Partidas	ELO medio	Victorias (%)	Partidas con blancas (%)
1	1000	1152	17,4	12,4
2	1000	1162	33,2	46,1
3	1000	1265	39,9	51,1
4	1000	1185	19,2	33
5	1000	1174	28	50
6	1000	1088	18,4	49,1
7	1000	1204	27,8	44,6
8	1000	1141	44,1	46,7
9	1000	1162	30	47,2
10	1000	1243	52	48,4
11	1000	1335	30,2	37,4
12	1000	1169	37,7	51,2
13	1000	1237	46	47,2
14	1000	1414	40,5	85,9
15	1000	1319	39,6	50,8
16	1000	1439	44,6	44,5
17	1000	1531	41,8	52,4
18	1000	1609	47,8	74,1
19	1000	1492	35,3	49,8
20	1000	1413	41,8	50,8
21	1000	1523	53,5	47,9
22	1000	1676	53,1	45,3
23	1000	1654	40,3	53,2
24	1000	1665	38,9	46,1
25	1000	1705	49,9	41,7
26	1000	1728	55	59,7
27	1000	1439	44,3	48,9
28	1000	1720	44,4	71,3
29	1000	1810	59,6	62,6
30	1000	1814	59,2	47,3
31	1000	1801	55,7	49,4
32	1000	1867	46,6	47,8
33	1000	2037	70,3	54,4
34	1000	1544	39,6	49,3
35	1000	1923	72,2	49,6
36	1000	1912	76,7	48,3

Tabla 4.5 - Información de los jugadores del conjunto 1000 (solo se tienen en cuenta datos del conjunto de entrenamiento)

Jugador	Partidas	ELO medio	Victorias (%)	Partidas con blancas (%)
1	500	1180	17,4	15,8
2	500	1168	33,4	48
3	500	1280	37,6	80
4	500	1203	20,6	32,8
5	500	1174	26,4	49
6	500	1109	20,2	48,8
7	500	1208	24	44,2
8	500	1103	37,2	46,8
9	500	1152	22,4	44,8
10	500	1243	55,2	50,8
11	500	1349	29,6	34
12	500	1136	31,2	53,2
13	500	1463	24,2	38,2
14	500	1436	41,4	85,8
15	500	1308	37,4	50,4
16	500	1527	48	44,2
17	500	1560	45,6	49,8
18	500	1621	46,8	74,2
19	500	1518	39,8	50
20	500	1468	41,4	51,4
21	500	1530	51	48
22	500	1681	52,8	45
23	500	1623	42,2	53
24	500	1651	41,2	45,2
25	500	1647	50,2	40
26	500	1738	59,4	60
27	500	1429	46,4	48,6
28	500	1732	49,8	81,2
29	500	1805	58,4	61,6
30	500	1831	59,4	47
31	500	1813	61	48,6
32	500	1876	51,2	46,2
33	500	2065	69,6	55
34	500	1520	38,2	49
35	500	1907	68,4	47,8
36	500	1893	75,6	46,6

Tabla 4.6 - Información de los jugadores del conjunto 500 (solo se tienen en cuenta datos del conjunto de entrenamiento)

Jugador	Partidas	ELO medio	Victorias (%)	Partidas con blancas (%)
1	250	1185	17,2	18
2	250	1164	30	47,2
3	250	1312	42,4	90,4
4	250	1214	21,2	30,8
5	250	1171	25,6	48,8
6	250	1109	20,2	48,8
7	250	1198	26,8	43,6
8	250	1048	23,2	46
9	250	1138	20	42,4
10	250	1260	57,6	52
11	250	1337	28,8	34,4
12	250	1140	26,4	58
13	250	1493	22,4	40,8
14	250	1449	43,2	83,6
15	250	1323	36,4	48,8
16	250	1522	49,6	43,6
17	250	1547	42,8	52
18	250	1647	45,2	76,4
19	250	1533	42	49,2
20	250	1530	44,8	51,2
21	250	1522	46,4	50
22	250	1688	48,4	46,8
23	250	1620	40	54,4
24	250	1624	39,2	46,4
25	250	1608	50	38,4
26	250	1729	56,4	59,6
27	250	1402	52,4	46
28	250	1711	53,2	84
29	250	1781	57,6	60,8
30	250	1849	60,8	46,4
31	250	1810	58,8	48
32	250	1896	52,4	45,6
33	250	2061	67,2	55,6
34	250	1504	37,6	50
35	250	1855	68	49,2
36	250	1926	78	48,4

Tabla 4.7 - Información de los jugadores del conjunto 250 (solo se tienen en cuenta datos del conjunto de entrenamiento)

Jugador	Partidas	ELO medio	Victorias (%)	Partidas con blancas (%)
1	100	1157	16	21
2	100	1180	25	43
3	100	1347	41	95
4	100	1224	23	30
5	100	1169	10	47
6	100	1175	25	48
7	100	1177	27	47
8	100	1106	24	46
9	100	1110	16	41
10	100	1266	62	54
11	100	1333	27	30
12	100	1157	25	76
13	100	1533	23	46
14	100	1450	43	77
15	100	1347	41	50
16	100	1604	58	48
17	100	1556	50	49
18	100	1672	47	77
19	100	1547	45	48
20	100	1566	38	49
21	100	1494	40	53
22	100	1753	53	49
23	100	1595	38	56
24	100	1631	44	47
25	100	1573	49	37
26	100	1735	54	60
27	100	1375	48	46
28	100	1735	59	95
29	100	1766	53	60
30	100	1845	63	51
31	100	1852	59	48
32	100	1911	56	46
33	100	2061	73	55
34	100	1478	34	46
35	100	1797	66	50
36	100	1924	77	46

Tabla 4.8 - Información de los jugadores del conjunto 100 (solo se tienen en cuenta datos del conjunto de entrenamiento)

Jugador	Partidas	ELO medio	Victorias (%)	Partidas con blancas (%)
1	50	1150	18	24
2	50	1198	26	44
3	50	1360	40	92
4	50	1227	28	30
5	50	1217	6	52
6	50	1190	28	44
7	50	1171	22	44
8	50	1145	18	42
9	50	1143	16	36
10	50	1241	56	56
11	50	1333	24	38
12	50	1195	22	78
13	50	1538	26	46
14	50	1510	38	68
15	50	1380	36	50
16	50	1621	60	52
17	50	1549	44	46
18	50	1696	44	84
19	50	1527	50	42
20	50	1599	36	52
21	50	1490	32	52
22	50	1735	48	48
23	50	1586	30	56
24	50	1570	36	50
25	50	1571	52	32
26	50	1747	50	64
27	50	1366	48	46
28	50	1685	66	100
29	50	1754	48	60
30	50	1832	58	52
31	50	1869	64	46
32	50	1899	56	44
33	50	2017	68	58
34	50	1435	28	44
35	50	1766	64	54
36	50	1902	78	44

Tabla 4.9 - Información de los jugadores del conjunto 50 (solo se tienen en cuenta datos del conjunto de entrenamiento)

Como se puede observar, al comparar estas tablas con la tabla 4.2, se pueden apreciar diferencias considerables. Para empezar, lo que más llama la atención es la diferencia en el ELO que existe entre ellas. La mayoría de los jugadores de los nuevos conjuntos tienen un ELO menor que los del conjunto base, y al comparar los nuevos conjuntos entre sí se puede ver que esta diferencia aumenta según se reduce el número de partidas. Esto era de esperar, puesto que se han eliminado miles de partidas dejando únicamente las más antiguas, es decir, se han borrado partidas en las que, por lo general, estos jugadores mejoraban su estilo de juego y, por tanto, conseguían un mayor rango.

Además, otro efecto de eliminar estas partidas ha sido la pérdida algunas de las condiciones impuestas para la selección de jugadores durante la creación del conjunto base, más concretamente las siguientes:

- El porcentaje de partidas jugadas con blancas equilibrado.
- Los jugadores han de pertenecer a uno de los rangos de ELO determinados por Maia.

En los nuevos conjuntos existen casos de jugadores que tienen un porcentaje de partidas jugadas con blancas demasiado alto, mucho mayor del que tenían en el conjunto base. Sin embargo, estos jugadores (3, 14, 18) tienen un porcentaje de victorias bastante equilibrado a lo largo de todos los conjuntos y no deberían afectar a los resultados. A pesar de esto, en el conjunto 50 se puede ver que el jugador 28 tiene un 100% de partidas jugadas con blancas, por lo que, al igual que pasó con el jugador 1 en el conjunto base, no podrá ser entrenado.

En cuanto a los rangos de ELO, el jugador 33 sigue por encima de los rangos utilizados por los desarrolladores de Maia, de hecho, en todos estos conjuntos tiene más ELO que el que tenía en el conjunto base. Además, en el conjunto 1000 el jugador 6 se encuentra por debajo del rango 1100. A pesar de esto, el ELO de ambos jugadores no difiere en exceso del establecido en Maia, por lo que no deberían suponer ningún problema.

Tras el entrenamiento de estos 5 conjuntos, se ha creado una gran cantidad de gráficas con los datos obtenidos, las cuales se pueden encontrar en el anexo. Al analizarlas, lo primero que se puede observar es que, a diferencia con el conjunto base, el jugador 1 ha podido ser entrenado en todos los conjuntos, ya que por fin dispone de suficientes partidas jugadas con

blancas en todos ellos. Sin embargo, como ya se ha dicho, en el conjunto 50 el jugador 28 no dispone de las suficientes partidas jugadas como negras, por lo que en este caso no se ha podido llevar a cabo su entrenamiento.

Por otra parte, al igual que sucedía con el conjunto base, durante los primeros 1000 pasos ya se puede observar una gran mejora en la precisión de las cabezas “policy” y “value”. De hecho, en la mayoría de los jugadores de estos conjuntos, las precisiones más altas se alcanzan en estos pasos. Esto no se debe a que la precisión durante los primeros pasos haya mejorado respecto a la del conjunto base, sino que la precisión obtenida durante el resto de los pasos ha empeorado, siendo incluso más inestable que antes.

De manera similar a lo que se ha hecho en el apartado anterior, para estudiar las precisiones obtenidas durante los primeros 1000 pasos se crearán 5 tablas, una por cada conjunto, que comparen la precisión inicial, es decir, la precisión de la red Maia-1900, con la precisión obtenida a los 100, 500 y 1000 pasos de entrenamiento. Además, todas estas tablas se compararán a su vez con la tabla 4.4, pudiendo de esta forma observar el efecto de la reducción de partidas durante los primeros pasos del entrenamiento.

A continuación se muestran las 5 tablas, referentes a los conjuntos 1000, 500, 250 y 50 respectivamente.

Jugador	P (0)	V (0)	P (100)	V (100)	MP (100)	MV (100)	P (500)	V (500)	MP (500)	MV (500)	P (1000)	V (1000)	MP (1000)	MV (1000)
1	37.1094	72.2656	40.2344	85.1562	3.125	12.8906	43.75	78.125	6.6406	5.8594	36.3281	76.9531	-0.7813	4.6875
2	39.8438	56.6406	39.8438	68.75	0	12.1094	37.5	65.2344	-2.3438	8.5938	42.5781	56.6406	2.7343	0
3	45.7031	45.7031	38.2812	75.7812	-7.4219	30.0781	39.0625	53.5156	-6.6406	7.8125	32.8125	52.7344	-12.8906	7.0313
4	35.5469	69.1406	43.3594	87.5	7.8125	18.3594	40.2344	80.4688	4.6875	11.3282	33.2031	81.6406	-2.3438	12.5
5	45.7031	56.6406	57.4219	57.0312	11.7188	0.3906	49.6094	76.5625	3.9063	19.9219	50.7812	62.5	5.0781	5.8594
6	33.2031	50	46.875	81.6406	13.6719	31.6406	37.8906	82.0312	4.6875	32.0312	38.2812	77.7344	5.0781	27.7344
7	46.4844	51.5625	53.9062	68.3594	7.4218	16.7969	53.125	57.0312	6.6406	5.4687	53.125	65.2344	6.6406	13.6719
8	48.0469	64.8438	48.8281	62.5	0.7812	-2.3438	55.8594	63.2812	7.8125	-1.5626	38.6719	59.7656	-9.375	-5.0782
9	37.5	62.8906	43.3594	70.3125	5.8594	7.4219	36.3281	76.5625	-1.1719	13.6719	35.1562	74.2188	-2.3438	11.3282
10	49.6094	50	50.3906	60.1562	0.7812	10.1562	50.7812	58.2031	1.1718	8.2031	45.7031	58.9844	-3.9063	8.9844
11	46.0938	49.2188	54.2969	67.1875	8.2031	17.9687	42.5781	61.3281	-3.5157	12.1093	44.9219	63.6719	-1.1719	14.4531
12	40.2344	72.2656	39.4531	62.5	-0.7813	-9.7656	36.3281	51.5625	-3.9063	-20.7031	35.5469	49.6094	-4.6875	-22.6562
13	49.6094	50.3906	48.8281	67.1875	-0.7813	16.7969	50	68.3594	0.3906	17.9688	47.6562	66.7969	-1.9532	16.4063
14	42.9688	57.0312	50	62.1094	7.0312	5.0782	48.8281	72.2656	5.8593	15.2344	51.9531	58.5938	8.9843	1.5626
15	47.2656	53.125	53.5156	54.2969	6.25	1.1719	52.3438	59.375	5.0782	6.25	50.3906	54.6875	3.125	1.5625
16	47.2656	45.7031	46.4844	57.4219	-0.7812	11.7188	45.3125	52.7344	-1.9531	7.0313	43.3594	51.9531	-3.9062	6.25
17	54.6875	58.2031	47.6562	57.8125	-7.0313	-0.3906	44.9219	59.7656	-9.7656	1.5625	50.7812	64.4531	-3.9063	6.25
18	48.4375	64.0625	45.7031	58.5938	-2.7344	-5.4687	35.9375	62.8906	-12.5	-1.1719	37.5	58.2031	-10.9375	-5.8594
19	62.8906	53.125	51.9531	68.3594	-10.9375	15.2344	44.9219	68.3594	-17.9687	15.2344	53.5156	65.625	-9.375	12.5
20	50	60.9375	51.1719	66.4062	1.1719	5.4687	46.875	59.7656	-3.125	-1.1719	46.0938	56.6406	-3.9062	-4.2969
21	37.5	60.1562	41.4062	63.6719	3.9062	3.5157	42.1875	53.5156	4.6875	-6.6406	39.8438	53.9062	2.3438	-6.25
22	50.7812	69.9219	56.6406	62.8906	5.8594	-7.0313	52.3438	50.7812	1.5626	-19.1407	55.0781	53.5156	4.2969	-16.4063
23	60.5469	50.3906	55.8594	71.0938	-4.6875	20.7032	54.6875	61.7188	-5.8594	11.3282	51.1719	58.9844	-9.375	8.5938
24	44.1406	61.3281	51.1719	60.9375	7.0313	-0.3906	51.1719	54.2969	7.0313	-7.0312	46.4844	48.4375	2.3438	-12.8906
25	54.2969	61.7188	45.3125	54.2969	-8.9844	-7.4219	39.0625	55.8594	-15.2344	-5.8594	44.5312	51.9531	-9.7657	-9.7657

26	47.6562	50.7812	50.3906	51.1719	2.7344	0.3907	50.3906	56.6406	2.7344	5.8594	48.0469	58.9844	0.3907	8.2032
27	45.7031	57.8125	50.7812	68.3594	5.0781	10.5469	49.2188	53.125	3.5157	-4.6875	41.0156	58.9844	-4.6875	1.1719
28	53.125	69.9219	57.4219	68.75	4.2969	-1.1719	54.6875	60.5469	1.5625	-9.375	47.2656	61.7188	-5.8594	-8.2031
29	55.4688	58.5938	49.2188	71.4844	-6.25	12.8906	52.3438	62.1094	-3.125	3.5156	51.5625	69.9219	-3.9063	11.3281
30	53.5156	67.9688	55.4688	57.0312	1.9532	-10.9376	51.9531	55.4688	-1.5625	-12.5	48.8281	52.3438	-4.6875	-15.625
31	55.0781	61.7188	57.0312	64.4531	1.9531	2.7343	54.6875	49.6094	-0.3906	-12.1094	50	48.8281	-5.0781	-12.8907
32	50	57.4219	52.7344	50.7812	2.7344	-6.6407	44.5312	59.375	-5.4688	1.9531	49.2188	51.9531	-0.7812	-5.4688
33	54.2969	55.0781	54.2969	71.0938	0	16.0157	50.3906	74.6094	-3.9063	19.5313	49.2188	55.4688	-5.0781	0.3907
34	45.3125	59.375	45.3125	54.2969	0	-5.0781	39.4531	58.5938	-5.8594	-0.7812	35.1562	46.875	-10.1563	-12.5
35	45.7031	74.2188	48.8281	84.7656	3.125	10.5468	55.0781	78.125	9.375	3.9062	55.8594	70.3125	10.1563	-3.9063
36	48.8281	60.9375	43.75	75	-5.0781	14.0625	49.6094	70.3125	0.7813	9.375	37.5	78.9062	-11.3281	17.9687

Tabla 4.10 - Mejora de precisión (“policy” y “value”) del conjunto 1000 respecto a Maia-1900 en los pasos de entrenamiento 100, 500 y 1000

Jugador	P (0)	V (0)	P (100)	V (100)	MP (100)	MV (100)	P (500)	V (500)	MP (500)	MV (500)	P (1000)	V (1000)	MP (1000)	MV (1000)
1	36,7188	73,4375	45,7031	88,2812	8,9843	14,8437	41,7969	81,25	5,0781	7,8125	36,3281	73,8281	-0,3907	0,3906
2	37,5	63,2812	46,4844	70,7031	8,9844	7,4219	39,4531	62,1094	1,9531	-1,1718	45,3125	58,2031	7,8125	-5,0781
3	47,2656	59,375	45,7031	64,0625	-1,5625	4,6875	38,6719	59,7656	-8,5937	0,3906	40,2344	59,375	-7,0312	0
4	39,4531	74,2188	44,1406	78,9062	4,6875	4,6874	39,0625	79,2969	-0,3906	5,0781	36,3281	77,7344	-3,125	3,5156
5	42,1875	58,9844	55,4688	57,8125	13,2813	-1,1719	52,7344	60,1562	10,5469	1,1718	53,9062	50,7812	11,7187	-8,2032
6	39,0625	57,0312	38,6719	82,4219	-0,3906	25,3907	34,7656	75,7812	-4,2969	18,75	37,1094	82,0312	-1,9531	25
7	50,7812	67,5781	49,2188	67,1875	-1,5624	-0,3906	43,3594	57,0312	-7,4218	-10,5469	43,75	67,1875	-7,0312	-0,3906
8	48,4375	71,0938	44,5312	58,9844	-3,9063	-12,1094	46,875	57,4219	-1,5625	-13,6719	41,7969	43,75	-6,6406	-27,3438
9	39,4531	76,1719	41,0156	60,1562	1,5625	-16,0157	35,9375	68,75	-3,5156	-7,4219	34,7656	66,7969	-4,6875	-9,375
10	53,125	53,125	51,5625	62,1094	-1,5625	8,9844	55,4688	58,5938	2,3438	5,4688	51,5625	60,1562	-1,5625	7,0312
11	51,1719	67,5781	57,4219	61,3281	6,25	-6,25	45,3125	70,7031	-5,8594	3,125	46,4844	55,0781	-4,6875	-12,5
12	42,9688	61,3281	40,2344	65,625	-2,7344	4,2969	43,75	55,0781	0,7812	-6,25	33,2031	53,125	-9,7657	-8,2031
13	49,6094	44,5312	49,2188	66,4062	-0,3906	21,875	44,9219	55,0781	-4,6875	10,5469	39,4531	68,3594	-10,1563	23,8282
14	39,0625	64,4531	48,0469	58,5938	8,9844	-5,8593	46,875	64,8438	7,8125	0,3907	46,875	60,5469	7,8125	-3,9062
15	45,3125	60,9375	49,2188	57,8125	3,9063	-3,125	51,9531	57,8125	6,6406	-3,125	51,5625	56,6406	6,25	-4,2969
16	55,0781	54,6875	51,1719	51,5625	-3,9062	-3,125	44,9219	51,9531	-10,1562	-2,7344	47,2656	57,0312	-7,8125	2,3437
17	49,6094	66,7969	55,0781	64,4531	5,4687	-2,3438	50,7812	69,5312	1,1718	2,7343	49,6094	61,3281	0	-5,4688
18	50,3906	63,6719	48,4375	65,2344	-1,9531	1,5625	41,0156	61,3281	-9,375	-2,3438	39,8438	61,7188	-10,5468	-1,9531
19	53,9062	45,7031	50,7812	71,4844	-3,125	25,7813	46,875	57,8125	-7,0312	12,1094	46,875	53,125	-7,0312	7,4219
20	55,4688	68,75	50,3906	64,0625	-5,0782	-4,6875	49,2188	58,5938	-6,25	-10,1562	41,7969	66,4062	-13,6719	-2,3438
21	41,0156	57,8125	44,9219	60,5469	3,9063	2,7344	38,6719	57,0312	-2,3437	-0,7813	35,1562	51,1719	-5,8594	-6,6406
22	49,6094	69,9219	48,0469	58,5938	-1,5625	-11,3281	47,6562	62,8906	-1,9532	-7,0313	50	60,9375	0,3906	-8,9844
23	54,6875	56,25	51,9531	56,6406	-2,7344	0,3906	53,125	45,7031	-1,5625	-10,5469	52,7344	51,1719	-1,9531	-5,0781
24	49,6094	47,6562	48,8281	54,2969	-0,7813	6,6407	51,5625	50	1,9531	2,3438	48,0469	54,6875	-1,5625	7,0313
25	53,125	70,7031	47,6562	60,9375	-5,4688	-9,7656	39,4531	56,25	-13,6719	-14,4531	42,9688	58,5938	-10,1562	-12,1093

26	50,3906	57,0312	53,9062	59,7656	3,5156	2,7344	44,9219	55,0781	-5,4687	-1,9531	42,1875	56,25	-8,2031	-0,7812
27	54,2969	56,25	51,9531	67,1875	-2,3438	10,9375	48,4375	53,9062	-5,8594	-2,3438	46,0938	53,9062	-8,2031	-2,3438
28	49,2188	58,9844	51,1719	71,0938	1,9531	12,1094	47,2656	59,375	-1,9532	0,3906	51,1719	54,6875	1,9531	-4,2969
29	50,7812	57,0312	55,4688	63,2812	4,6876	6,25	44,5312	58,9844	-6,25	1,9532	48,0469	55,8594	-2,7343	-1,1718
30	55,8594	58,5938	57,0312	63,6719	1,1718	5,0781	47,6562	52,7344	-8,2032	-5,8594	54,2969	44,5312	-1,5625	-14,0626
31	55,4688	56,25	64,0625	64,4531	8,5937	8,2031	46,0938	54,6875	-9,375	-1,5625	52,3438	63,2812	-3,125	7,0312
32	46,875	59,7656	47,6562	53,9062	0,7812	-5,8594	47,6562	50,7812	0,7812	-8,9844	45,7031	58,9844	-1,1719	-0,7812
33	59,375	62,1094	51,9531	72,6562	-7,4219	10,5468	53,9062	61,7188	-5,4688	-0,3906	53,5156	67,1875	-5,8594	5,0781
34	44,5312	60,5469	42,1875	59,375	-2,3437	-1,1719	36,3281	56,25	-8,2031	-4,2969	37,5	44,5312	-7,0312	-16,0157
35	47,2656	70,7031	50,7812	68,3594	3,5156	-2,3437	46,875	66,4062	-0,3906	-4,2969	45,3125	67,1875	-1,9531	-3,5156
36	53,9062	64,8438	43,75	75,3906	-10,1562	10,5468	43,75	66,0156	-10,1562	1,1718	43,3594	70,7031	-10,5468	5,8593

Tabla 4.11 - Mejora de precisión (“policy” y “value”) del conjunto 500 respecto a Maia-1900 en los pasos de entrenamiento 100, 500 y 1000

Jugador	P (0)	V (0)	P (100)	V (100)	MP (100)	MV (100)	P (500)	V (500)	MP (500)	MV (500)	P (1000)	V (1000)	MP (1000)	MV (1000)
1	39,8438	70,3125	48,0469	82,8125	8,2031	12,5	46,0938	78,9062	6,25	8,5937	44,9219	73,0469	5,0781	2,7344
2	38,2812	58,5938	40,2344	67,1875	1,9532	8,5937	38,2812	63,2812	0	4,6874	38,6719	60,5469	0,3907	1,9531
3	48,8281	61,7188	44,9219	46,0938	-3,9062	-15,625	31,6406	57,0312	-17,1875	-4,6876	39,4531	56,6406	-9,375	-5,0782
4	40,2344	55,0781	41,4062	73,4375	1,1718	18,3594	37,1094	68,75	-3,125	13,6719	38,2812	70,7031	-1,9532	15,625
5	49,2188	51,5625	57,0312	70,3125	7,8124	18,75	46,0938	65,2344	-3,125	13,6719	46,875	69,9219	-2,3438	18,3594
6	41,7969	69,1406	40,625	78,9062	-1,1719	9,7656	35,1562	77,7344	-6,6407	8,5938	37,1094	71,875	-4,6875	2,7344
7	46,0938	50,7812	50,7812	66,4062	4,6874	15,625	50,7812	56,25	4,6874	5,4688	50	62,5	3,9062	11,7188
8	48,4375	66,7969	43,3594	56,25	-5,0781	-10,5469	39,0625	55,4688	-9,375	-11,3281	46,4844	50,7812	-1,9531	-16,0157
9	36,3281	58,9844	36,7188	80,0781	0,3907	21,0937	37,1094	70,7031	0,7813	11,7187	34,7656	62,8906	-1,5625	3,9062
10	49,2188	62,1094	47,6562	51,9531	-1,5626	-10,1563	51,9531	51,1719	2,7343	-10,9375	53,9062	53,125	4,6874	-8,9844
11	50,3906	61,7188	50,7812	56,6406	0,3906	-5,0782	47,2656	67,1875	-3,125	5,4687	39,0625	68,3594	-11,3281	6,6406
12	40,625	54,2969	38,6719	59,375	-1,9531	5,0781	40,2344	65,2344	-0,3906	10,9375	38,2812	63,2812	-2,3438	8,9843
13	52,7344	60,9375	56,6406	64,8438	3,9062	3,9063	50,7812	60,5469	-1,9532	-0,3906	55,0781	65,625	2,3437	4,6875
14	41,4062	58,2031	54,2969	52,3438	12,8907	-5,8593	44,1406	66,4062	2,7344	8,2031	47,2656	53,125	5,8594	-5,0781
15	56,25	68,75	58,5938	57,0312	2,3438	-11,7188	52,7344	58,5938	-3,5156	-10,1562	53,9062	57,4219	-2,3438	-11,3281
16	55,4688	61,3281	47,2656	63,2812	-8,2032	1,9531	47,2656	59,7656	-8,2032	-1,5625	44,9219	50,7812	-10,5469	-10,5469
17	45,7031	67,1875	46,4844	64,0625	0,7813	-3,125	45,3125	51,1719	-0,3906	-16,0156	47,6562	58,5938	1,9531	-8,5937
18	48,0469	62,1094	53,9062	62,5	5,8593	0,3906	48,0469	58,5938	0	-3,5156	44,5312	57,0312	-3,5157	-5,0782
19	62,8906	53,125	45,7031	64,8438	-17,1875	11,7188	50	53,5156	-12,8906	0,3906	42,5781	56,25	-20,3125	3,125
20	53,125	62,8906	50,7812	60,9375	-2,3438	-1,9531	45,7031	60,1562	-7,4219	-2,7344	46,4844	57,8125	-6,6406	-5,0781
21	40,2344	50,3906	42,1875	54,6875	1,9531	4,2969	39,8438	54,6875	-0,3906	4,2969	37,5	60,9375	-2,7344	10,5469
22	55,4688	67,1875	51,5625	55,0781	-3,9063	-12,1094	52,3438	60,1562	-3,125	-7,0313	50	62,5	-5,4688	-4,6875
23	59,375	61,3281	54,2969	62,8906	-5,0781	1,5625	53,9062	53,9062	-5,4688	-7,4219	54,2969	54,6875	-5,0781	-6,6406
24	52,7344	62,1094	48,8281	52,7344	-3,9063	-9,375	54,6875	46,4844	1,9531	-15,625	45,3125	53,9062	-7,4219	-8,2032
25	47,2656	67,5781	40,625	71,875	-6,6406	4,2969	44,1406	62,1094	-3,125	-5,4687	42,9688	56,6406	-4,2968	-10,9375

26	51,5625	68,3594	46,0938	44,9219	-5,4687	-23,4375	46,0938	54,2969	-5,4687	-14,0625	50,3906	56,6406	-1,1719	-11,7188
27	53,5156	48,8281	50,3906	57,0312	-3,125	8,2031	51,5625	48,4375	-1,9531	-0,3906	51,9531	46,4844	-1,5625	-2,3437
28	49,2188	59,375	53,5156	58,5938	4,2968	-0,7812	48,0469	53,5156	-1,1719	-5,8594	49,6094	56,25	0,3906	-3,125
29	47,2656	53,9062	46,0938	50,7812	-1,1718	-3,125	49,6094	53,5156	2,3438	-0,3906	49,6094	51,1719	2,3438	-2,7343
30	57,8125	68,3594	51,9531	62,8906	-5,8594	-5,4688	54,2969	60,5469	-3,5156	-7,8125	50,7812	62,5	-7,0313	-5,8594
31	55,8594	77,7344	53,5156	63,6719	-2,3438	-14,0625	51,1719	53,125	-4,6875	-24,6094	54,2969	57,4219	-1,5625	-20,3125
32	57,0312	55,4688	48,0469	44,9219	-8,9843	-10,5469	47,2656	45,3125	-9,7656	-10,1563	46,875	54,6875	-10,1562	-0,7813
33	64,0625	63,6719	53,5156	65,625	-10,5469	1,9531	51,9531	62,1094	-12,1094	-1,5625	60,1562	59,7656	-3,9063	-3,9063
34	42,1875	59,7656	42,5781	66,4062	0,3906	6,6406	39,4531	50,3906	-2,7344	-9,375	37,5	53,9062	-4,6875	-5,8594
35	50,3906	53,9062	48,0469	55,0781	-2,3437	1,1719	52,3438	56,25	1,9532	2,3438	49,6094	51,5625	-0,7812	-2,3437
36	50	69,1406	46,4844	75,3906	-3,5156	6,25	47,6562	68,75	-2,3438	-0,3906	47,6562	71,875	-2,3438	2,7344

Tabla 4.12 - Mejora de precisión (“policy” y “value”) del conjunto 250 respecto a Maia-1900 en los pasos de entrenamiento 100, 500 y 1000

Jugador	P (0)	V (0)	P (100)	V (100)	MP (100)	MV (100)	P (500)	V (500)	MP (500)	MV (500)	P (1000)	V (1000)	MP (1000)	MV (1000)
1	40,2344	71,4844	44,9219	84,7656	4,6875	13,2812	44,1406	80,0781	3,9062	8,5937	41,0156	76,1719	0,7812	4,6875
2	43,3594	58,9844	42,9688	55,8594	-0,3906	-3,125	39,4531	58,9844	-3,9063	0	46,875	64,0625	3,5156	5,0781
3	42,1875	64,4531	39,8438	68,75	-2,3437	4,2969	41,4062	61,3281	-0,7813	-3,125	41,7969	52,3438	-0,3906	-12,1093
4	35,5469	64,8438	33,9844	74,2188	-1,5625	9,375	36,7188	76,1719	1,1719	11,3281	33,2031	73,0469	-2,3438	8,2031
5	42,5781	53,9062	37,5	66,4062	-5,0781	12,5	37,5	65,2344	-5,0781	11,3282	37,8906	71,0938	-4,6875	17,1876
6	41,7969	71,0938	44,1406	77,7344	2,3437	6,6406	35,9375	83,2031	-5,8594	12,1093	39,4531	77,3438	-2,3438	6,25
7	45,7031	56,25	46,0938	72,6562	0,3907	16,4062	38,2812	69,5312	-7,4219	13,2812	48,0469	62,1094	2,3438	5,8594
8	35,1562	44,5312	35,9375	75,7812	0,7813	31,25	34,7656	68,3594	-0,3906	23,8282	39,0625	68,3594	3,9063	23,8282
9	41,0156	58,5938	42,5781	77,7344	1,5625	19,1406	34,7656	73,0469	-6,25	14,4531	41,0156	74,2188	0	15,625
10	42,9688	62,8906	48,4375	60,9375	5,4687	-1,9531	57,8125	57,0312	14,8437	-5,8594	48,0469	58,9844	5,0781	-3,9062
11	46,875	73,0469	48,0469	74,6094	1,1719	1,5625	48,4375	74,2188	1,5625	1,1719	43,3594	75	-3,5156	1,9531
12	44,1406	52,3438	40,625	61,7188	-3,5156	9,375	41,7969	60,9375	-2,3437	8,5937	34,7656	59,375	-9,375	7,0312
13	49,6094	56,25	48,0469	73,0469	-1,5625	16,7969	44,1406	65,625	-5,4688	9,375	43,75	66,4062	-5,8594	10,1562
14	44,9219	57,0312	53,5156	54,6875	8,5937	-2,3437	50,7812	50,7812	5,8593	-6,25	45,3125	46,0938	0,3906	-10,9374
15	45,7031	62,8906	53,5156	59,7656	7,8125	-3,125	49,6094	62,5	3,9063	-0,3906	55,8594	53,9062	10,1563	-8,9844
16	54,2969	45,3125	42,9688	57,0312	-11,3281	11,7187	44,1406	57,0312	-10,1563	11,7187	42,5781	50	-11,7188	4,6875
17	47,2656	60,1562	51,1719	65,625	3,9063	5,4688	51,1719	55,0781	3,9063	-5,0781	51,5625	46,4844	4,2969	-13,6718
18	44,5312	57,8125	46,4844	60,5469	1,9532	2,7344	41,7969	62,8906	-2,7343	5,0781	43,3594	55,8594	-1,1718	-1,9531
19	61,3281	49,2188	48,8281	51,5625	-12,5	2,3437	55,0781	51,5625	-6,25	2,3437	53,125	46,0938	-8,2031	-3,125
20	44,1406	61,3281	43,75	58,5938	-0,3906	-2,7343	41,0156	57,4219	-3,125	-3,9062	50,3906	59,7656	6,25	-1,5625
21	41,0156	60,1562	41,4062	63,6719	0,3906	3,5157	38,6719	53,9062	-2,3437	-6,25	43,75	54,2969	2,7344	-5,8593
22	50,7812	66,7969	56,25	58,5938	5,4688	-8,2031	53,5156	61,7188	2,7344	-5,0781	54,2969	62,1094	3,5157	-4,6875
23	57,4219	60,9375	56,25	60,1562	-1,1719	-0,7813	55,4688	55,0781	-1,9531	-5,8594	53,9062	56,25	-3,5157	-4,6875
24	50,3906	52,7344	48,8281	51,1719	-1,5625	-1,5625	48,0469	52,7344	-2,3437	0	48,4375	51,9531	-1,9531	-0,7813
25	52,7344	66,4062	44,5312	46,4844	-8,2032	-19,9218	51,1719	55,0781	-1,5625	-11,3281	41,0156	59,375	-11,7188	-7,0312

26	48,4375	67,5781	45,3125	56,6406	-3,125	-10,9375	48,4375	51,9531	0	-15,625	45,3125	58,5938	-3,125	-8,9843
27	46,875	60,9375	52,3438	64,4531	5,4688	3,5156	56,25	51,9531	9,375	-8,9844	48,4375	51,5625	1,5625	-9,375
28	50,7812	62,1094	48,4375	58,2031	-2,3437	-3,9063	57,8125	50,7812	7,0313	-11,3282	54,2969	52,3438	3,5157	-9,7656
29	53,5156	67,5781	42,5781	51,5625	-10,9375	-16,0156	48,8281	63,2812	-4,6875	-4,2969	45,7031	58,9844	-7,8125	-8,5937
30	57,4219	52,7344	53,5156	44,5312	-3,9063	-8,2032	56,6406	59,7656	-0,7813	7,0312	57,8125	54,2969	0,3906	1,5625
31	55,4688	64,0625	58,2031	58,9844	2,7343	-5,0781	51,9531	52,7344	-3,5157	-11,3281	51,9531	55,4688	-3,5157	-8,5937
32	48,8281	70,7031	55,0781	56,6406	6,25	-14,0625	45,7031	55,4688	-3,125	-15,2343	46,875	51,1719	-1,9531	-19,5312
33	59,375	56,6406	53,9062	60,9375	-5,4688	4,2969	53,9062	53,125	-5,4688	-3,5156	47,2656	62,1094	-12,1094	5,4688
34	39,4531	56,6406	40,625	58,2031	1,1719	1,5625	41,0156	43,3594	1,5625	-13,2812	33,9844	48,8281	-5,4687	-7,8125
35	51,5625	71,875	45,3125	44,1406	-6,25	-27,7344	55,0781	64,4531	3,5156	-7,4219	46,4844	58,2031	-5,0781	-13,6719
36	45,7031	54,6875	47,6562	66,4062	1,9531	11,7187	45,7031	60,9375	0	6,25	46,875	69,5312	1,1719	14,8437

Tabla 4.13 - Mejora de precisión (“policy” y “value”) del conjunto 100 respecto a Maia-1900 en los pasos de entrenamiento 100, 500 y 1000

Jugador	P (0)	V (0)	P (100)	V (100)	MP (100)	MV (100)	P (500)	V (500)	MP (500)	MV (500)	P (1000)	V (1000)	MP (1000)	MV (1000)
1	39,8438	67,1875	46,4844	82,0312	6,6406	14,8437	38,6719	74,2188	-1,1719	7,0313	37,8906	75	-1,9532	7,8125
2	44,1406	52,3438	41,7969	69,1406	-2,3437	16,7968	37,8906	71,0938	-6,25	18,75	33,9844	62,8906	-10,1562	10,5468
3	47,2656	52,3438	36,3281	62,5	-10,9375	10,1562	40,2344	67,1875	-7,0312	14,8437	35,1562	60,5469	-12,1094	8,2031
4	38,6719	69,5312	36,3281	81,25	-2,3438	11,7188	33,2031	73,8281	-5,4688	4,2969	28,5156	72,2656	-10,1563	2,7344
5	42,5781	63,2812	35,5469	64,4531	-7,0312	1,1719	35,9375	66,4062	-6,6406	3,125	32,4219	71,4844	-10,1562	8,2032
6	42,5781	73,4375	49,6094	81,25	7,0313	7,8125	40,625	78,5156	-1,9531	5,0781	41,0156	78,125	-1,5625	4,6875
7	42,9688	58,9844	45,3125	68,3594	2,3437	9,375	47,2656	73,4375	4,2968	14,4531	46,875	71,0938	3,9062	12,1094
8	35,5469	46,4844	39,8438	60,1562	4,2969	13,6718	38,2812	62,5	2,7343	16,0156	37,5	66,4062	1,9531	19,9218
9	37,1094	49,6094	40,625	74,2188	3,5156	24,6094	33,5938	71,4844	-3,5156	21,875	33,5938	73,0469	-3,5156	23,4375
10	46,875	56,6406	52,7344	59,375	5,8594	2,7344	51,9531	56,6406	5,0781	0	55,8594	48,8281	8,9844	-7,8125
11	44,9219	66,7969	50,7812	78,125	5,8593	11,3281	53,5156	73,4375	8,5937	6,6406	47,2656	66,4062	2,3437	-0,3907
12	41,7969	68,3594	44,1406	72,6562	2,3437	4,2968	36,7188	68,75	-5,0781	0,3906	33,2031	69,1406	-8,5938	0,7812
13	58,9844	52,3438	46,875	73,4375	-12,1094	21,0937	51,1719	67,9688	-7,8125	15,625	49,2188	66,7969	-9,7656	14,4531
14	35,1562	62,8906	44,9219	50,3906	9,7657	-12,5	44,5312	54,6875	9,375	-8,2031	48,0469	51,5625	12,8907	-11,3281
15	46,0938	60,1562	57,0312	46,4844	10,9374	-13,6718	51,1719	60,5469	5,0781	0,3907	51,9531	55,4688	5,8593	-4,6874
16	53,5156	48,4375	52,3438	62,5	-1,1718	14,0625	48,0469	53,125	-5,4687	4,6875	49,2188	53,125	-4,2968	4,6875
17	49,2188	66,4062	50	53,9062	0,7812	-12,5	51,9531	66,4062	2,7343	0	47,2656	58,5938	-1,9532	-7,8124
18	50	62,5	48,8281	59,7656	-1,1719	-2,7344	49,6094	62,5	-0,3906	0	45,3125	61,3281	-4,6875	-1,1719
19	55,8594	57,8125	48,0469	60,1562	-7,8125	2,3437	55,8594	60,5469	0	2,7344	50	53,9062	-5,8594	-3,9063
20	50,7812	61,7188	46,875	54,6875	-3,9062	-7,0313	43,3594	58,9844	-7,4218	-2,7344	42,9688	62,8906	-7,8124	1,1718
21	38,2812	60,5469	40,2344	55,0781	1,9532	-5,4688	37,8906	51,1719	-0,3906	-9,375	33,9844	50,3906	-4,2968	-10,1563
22	51,5625	66,7969	62,5	56,25	10,9375	-10,5469	48,0469	63,2812	-3,5156	-3,5157	54,6875	62,1094	3,125	-4,6875
23	55,8594	68,75	50,7812	48,8281	-5,0782	-19,9219	56,25	48,0469	0,3906	-20,7031	52,7344	53,9062	-3,125	-14,8438
24	50,7812	53,125	46,875	49,6094	-3,9062	-3,5156	51,5625	56,25	0,7813	3,125	50	55,4688	-0,7812	2,3438
25	55,8594	73,0469	46,875	60,1562	-8,9844	-12,8907	44,9219	60,1562	-10,9375	-12,8907	42,1875	58,5938	-13,6719	-14,4531

26	51,5625	55,4688	46,875	54,6875	-4,6875	-0,7813	48,0469	58,2031	-3,5156	2,7343	46,4844	50	-5,0781	-5,4688
27	57,8125	61,3281	54,2969	60,5469	-3,5156	-0,7812	54,2969	57,4219	-3,5156	-3,9062	51,9531	52,3438	-5,8594	-8,9843
28	-	-	-	-	-	-	-	-	-	-	-	-	-	-
29	49,6094	60,5469	51,1719	52,7344	1,5625	-7,8125	50	49,6094	0,3906	-10,9375	49,2188	56,25	-0,3906	-4,2969
30	59,375	70,3125	55,4688	48,0469	-3,9062	-22,2656	58,2031	52,3438	-1,1719	-17,9687	61,3281	59,375	1,9531	-10,9375
31	58,9844	70,3125	58,2031	65,2344	-0,7813	-5,0781	51,9531	60,5469	-7,0313	-9,7656	51,5625	54,2969	-7,4219	-16,0156
32	49,2188	57,4219	51,5625	41,4062	2,3437	-16,0157	44,9219	44,5312	-4,2969	-12,8907	44,9219	48,0469	-4,2969	-9,375
33	62,5	59,375	54,2969	41,4062	-8,2031	-17,9688	52,3438	62,5	-10,1562	3,125	50,3906	57,8125	-12,1094	-1,5625
34	44,1406	58,9844	39,8438	58,2031	-4,2968	-0,7813	39,4531	48,0469	-4,6875	-10,9375	33,2031	54,2969	-10,9375	-4,6875
35	48,8281	51,5625	46,875	50,3906	-1,9531	-1,1719	52,7344	57,4219	3,9063	5,8594	53,125	49,2188	4,2969	-2,3437
36	43,75	73,0469	47,6562	78,125	3,9062	5,0781	39,8438	76,5625	-3,9062	3,5156	41,0156	67,9688	-2,7344	-5,0781

Tabla 4.14 - Mejora de precisión (“policy” y “value”) del conjunto 50 respecto a Maia-1900 en los pasos de entrenamiento 100, 500 y 1000

Como era de esperar, al ir reduciendo el número de partidas utilizadas para el entrenamiento, también se va reduciendo la precisión de las redes neuronales. Sin embargo, se puede ver que existen excepciones. Por ejemplo, en el conjunto 500 se alcanzan las peores precisiones a los 1000 pasos, mucho menores que las obtenidas en el conjunto 50 utilizando los mismos pasos e incluso menos. De hecho, las precisiones obtenidas en el conjunto 50 utilizando 100 pasos son algo mejores. Esto puede deberse a que los conjuntos de test de los jugadores en el conjunto 50 estén formados por partidas que reflejen más el estilo que mostraron estos en las partidas del conjunto de entrenamiento, ya que no solo se obtienen peores resultados con 1000 pasos, sino que las precisiones con 100 y 500 también son menores que las obtenidas en conjuntos con menos partidas. Sin embargo, con los datos que se han extraído de las partidas resulta imposible conocer la causa exacta.

Por otra parte, en las tablas se puede ver que la precisión de “policy” y “value” suele ser peor en todos los conjuntos si se utilizan 1000 pasos de entrenamiento, obteniéndose, por lo general, mejores valores al emplear 500 e incluso 100 pasos. Esto puede deberse a que, al disminuir el número de partidas, la red neuronal esté viendo la totalidad del conjunto de entrenamiento más veces de las que lo hacía cuando este número era mayor. Por tanto, cuantos más pasos se empleen, más grande es el riesgo de que se produzca un sobreajuste de la red.

Por último, al comparar los valores de precisión obtenidos del conjunto base con los conseguidos durante este apartado, se puede ver que, por lo general, los nuevos valores son mucho más pequeños, llegando a haber diferencias de incluso 20 puntos porcentuales. Sin embargo, también es cierto que las precisiones alcanzadas por Maia-1900 en el conjunto base también son, por lo general, mayores. Estas precisiones dependen exclusivamente de las partidas de test seleccionadas, no del número de partidas utilizado para el entrenamiento, por lo que podrían indicar que los conjuntos de test utilizados en el conjunto base son más sencillos de predecir que los utilizados a lo largo de este apartado, lo que puede tener sentido ya que el estilo de juego de un jugador con muchas partidas a sus espaldas debería ser más estable que el de uno que tenga pocas. De todas formas, las diferencias entre estos valores no son muy significativas, por lo que no debería ser la razón de las bajas precisiones obtenidas durante el entrenamiento.

En conclusión, las precisiones obtenidas al reducir el número de partidas son peores que las obtenidas en el conjunto base. Esto se debe principalmente a la dificultad que tiene la red neuronal para extraer y generalizar el estilo de juego de un jugador a partir de un número de partidas muy pequeño. Sin embargo, observando las gráficas se puede ver que los valores de precisión obtenidos son muy inestables a lo largo de todo el entrenamiento, lo cual puede haberse producido al haber utilizado un valor de learning rate demasiado alto. Por esta razón, en el siguiente apartado se intentará buscar una solución desde esta perspectiva.

4.5. CAMBIO DE LEARNING RATE

A lo largo de todas las pruebas anteriores siempre se ha utilizado el mismo learning rate, el cual ha sido obtenido de la configuración por defecto de Maia individual. Como ya se ha dicho, este es un learning rate variable, cuyo valor va cambiando según avanza el entrenamiento de la red neuronal, tal y como aparece indicado en la tabla 4.3.

Como se puede ver en dicha tabla, este learning rate está diseñado para utilizarse en entrenamientos que utilicen 30000 pasos. Sin embargo, en las pruebas realizadas se han usado 10000, de los cuales, debido a los objetivos de este trabajo, solo resultan viables los 1000 primeros. Esto supone un problema, ya que el learning rate utilizado para estos pasos, además de ser fijo (ya que el primer cambio de valor ocurre a los 15000 pasos), es muy agresivo. Como se ha explicado al final del apartado anterior, este valor puede estar afectando a las precisiones de “policy” y “value” obtenidas en los entrenamientos de todos los conjuntos y, por tanto, influenciando negativamente en los en el proceso de creación de las IAs. Por este motivo, con el objetivo de suavizar el proceso de entrenamiento, se utilizará un learning rate que se ajuste a los 1000 pasos de entrenamiento buscados en este trabajo.

Al igual que el learning rate empleado por Maia individual, este también será variable, de forma que se utilicen valores más agresivos durante los primeros pasos y se reduzcan según avance el entrenamiento. Este nuevo learning rate aparece reflejado en la siguiente tabla:

Pasos de entrenamiento	Learning rate
[0, 100]	0,01
(100, 300]	0,001
(300, 500]	0,0001
(500, 1000]	0,00001

Tabla 4.15 - Nuevo learning rate, ajustado para utilizarse con 1000 pasos de entrenamiento

Es importante mencionar que no se ha realizado ningún estudio específico para determinar el valor del learning rate ni los pasos en los que este se modifica, por lo que muy probablemente esta no sea la solución óptima. Sin embargo, debería ser mejor que el learning rate anterior.

Para poder comparar los resultados obtenidos al usar este nuevo learning rate con los conseguidos en las pruebas del apartado anterior, es necesario utilizar los mismos conjuntos. Por tanto, se repetirán los entrenamientos de los conjuntos 1000, 500, 250, 100 y 50, pero haciendo uso de este nuevo learning rate. No obstante, a diferencia de las pruebas del conjunto anterior, el proceso de entrenamiento solo estará formado por 1000 pasos, ya que el nuevo learning rate está pensado para usar este número, además de que el tiempo necesario para entrenar a todos los jugadores de los 5 conjuntos es demasiado elevado.

Al igual que con las pruebas anteriores, tras el entrenamiento de todos los conjuntos se ha creado un gran número de gráficas con los datos obtenidos, las cuales se pueden encontrar en el anexo. Además, como en el apartado anterior, también se han creado 5 tablas de precisiones, una por cada conjunto entrenado. No obstante, a diferencia de las tablas anteriores, en estas no aparecen reflejadas las precisiones de “policy” y “value” obtenidas a los 100 pasos de entrenamiento. Esto se debe a que el valor de learning rate utilizado durante estos pasos es el mismo que se empleó en las pruebas anteriores, por lo que no tiene sentido volver a estudiarlo. Otra diferencia con las tablas del apartado anterior es la adición de nuevas columnas, las cuales hacen referencia al paso entrenamiento donde se alcanza el mínimo error total (“met”).

Como ya se ha dicho, una de las métricas obtenidas del conjunto de entrenamiento es el error total de la red, el cual se busca minimizar durante el proceso de entrenamiento. Este valor

puede servir para elegir la mejor configuración de pesos para la red, evitando elegir un número de pasos de manera manual. Tan solo es necesario almacenar los pesos de esta cada cierto número de pasos, tarea que ya realiza el programa de entrenamiento, y elegir los que produzcan el error total más bajo. Por este motivo, en las siguientes tablas se mostrarán las precisiones obtenidas al utilizar el número de pasos en los que se alcanza el mínimo error total de la red, denominados como “met”.

Jugador	P (0)	V (0)	P (500)	V (500)	MP (500)	MV (500)	P (1000)	V (1000)	MP (1000)	MV (1000)	P (met)	V (met)	MP (met)	MV (met)
1	41,0156	74,6094	50,39	72,27	9,375	-2,3438	45,7031	82,4219	4,6875	7,8125	46,4844	80,078	5,4688	5,4687
2	36,7188	59,7656	47,66	69,92	10,9374	10,1563	38,2812	70,3125	1,5624	10,5469	44,9219	69,141	8,2031	9,375
3	47,2656	53,9062	42,58	68,75	-4,6875	14,8438	49,6094	62,1094	2,3438	8,2032	45,3125	57,031	-1,9531	3,125
4	43,3594	75,3906	44,14	79,3	0,7812	3,9063	43,75	77,3438	0,3906	1,9532	41,7969	86,719	-1,5625	11,3282
5	43,3594	58,2031	53,91	73,05	10,5468	14,8438	55,0781	62,1094	11,7187	3,9063	57,4219	64,063	14,0625	5,8594
6	30,8594	52,7344	47,66	84,77	16,7968	32,0312	46,875	88,6719	16,0156	35,9375	44,5312	82,422	13,6718	29,6875
7	45,3125	44,9219	51,17	64,45	5,8594	19,5312	56,25	69,1406	10,9375	24,2187	54,6875	69,922	9,375	25
8	52,7344	69,5312	56,25	67,58	3,5156	-1,9531	54,6875	65,2344	1,9531	-4,2968	57,8125	66,797	5,0781	-2,7343
9	38,2812	62,5	46,09	78,13	7,8126	15,625	46,0938	77,3438	7,8126	14,8438	47,6562	69,922	9,375	7,4219
10	43,3594	54,6875	52,34	53,52	8,9844	-1,1719	62,1094	58,2031	18,75	3,5156	62,8906	58,203	19,5312	3,5156
11	41,4062	49,6094	50,78	70,31	9,375	20,7031	48,4375	71,875	7,0313	22,2656	50,7812	70,313	9,375	20,7031
12	43,75	70,3125	52,73	60,55	8,9844	-9,7656	45,7031	58,5938	1,9531	-11,7187	44,1406	55,078	0,3906	-15,234
13	52,3438	56,25	57,03	72,66	4,6874	16,4062	55,4688	73,8281	3,125	17,5781	53,125	66,797	0,7812	10,5469
14	48,0469	53,125	53,13	62,11	5,0781	8,9844	52,3438	69,5312	4,2969	16,4062	53,125	62,109	5,0781	8,9844
15	46,4844	60,5469	57,03	59,77	10,5468	-0,7813	55,4688	56,25	8,9844	-4,2969	55,8594	59,766	9,375	-0,7813
16	50,3906	53,5156	54,69	64,84	4,2969	11,3282	51,9531	61,3281	1,5625	7,8125	46,875	65,625	-3,5156	12,1094
17	51,5625	60,1562	53,52	60,16	1,9531	0	57,0312	61,7188	5,4687	1,5626	57,8125	61,719	6,25	1,5626
18	52,7344	67,1875	50,78	64,45	-1,9532	-2,7344	48,4375	64,4531	-4,2969	-2,7344	46,875	63,672	-5,8594	-3,5156
19	62,1094	53,125	55,47	68,75	-6,6406	15,625	60,5469	68,75	-1,5625	15,625	60,5469	68,75	-1,5625	15,625
20	50,7812	62,5	56,64	68,75	5,8594	6,25	48,0469	64,4531	-2,7343	1,9531	42,9688	63,672	-7,8124	1,1719
21	39,8438	63,6719	41,41	63,28	1,5624	-0,3907	47,2656	60,5469	7,4218	-3,125	46,875	60,156	7,0312	-3,5157
22	54,2969	66,7969	55,08	58,59	0,7812	-8,2031	53,5156	62,5	-0,7813	-4,2969	66,4062	64,063	12,1093	-2,7344
23	57,0312	52,7344	56,25	65,23	-0,7812	12,5	62,1094	70,3125	5,0782	17,5781	61,7188	66,797	4,6876	14,0625
24	56,25	64,0625	59,77	58,2	3,5156	-5,8594	52,7344	62,8906	-3,5156	-1,1719	51,9531	63,281	-4,2969	-0,7813
25	57,0312	66,4062	52,34	60,94	-4,6874	-5,4687	56,25	55,4688	-0,7812	-10,9374	50,3906	56,25	-6,6406	-10,156

26	46,875	46,875	60,16	62,5	13,2812	15,625	57,4219	53,125	10,5469	6,25	54,2969	48,828	7,4219	1,9531
27	51,1719	58,9844	57,81	60,94	6,6406	1,9531	49,2188	60,1562	-1,9531	1,1718	49,6094	69,141	-1,5625	10,1562
28	50,3906	64,8438	57,81	66,02	7,4219	1,1718	60,5469	59,7656	10,1563	-5,0782	56,6406	66,406	6,25	1,5624
29	50,3906	53,9062	58,98	60,94	8,5938	7,0313	65,625	67,5781	15,2344	13,6719	59,7656	62,891	9,375	8,9844
30	57,8125	68,3594	58,59	61,33	0,7813	-7,0313	57,4219	63,2812	-0,3906	-5,0782	58,9844	63,672	1,1719	-4,6875
31	61,3281	67,9688	61,72	59,77	0,3907	-8,2032	60,1562	61,3281	-1,1719	-6,6407	64,8438	58,594	3,5157	-9,375
32	51,5625	67,9688	48,83	56,64	-2,7344	-11,328	56,25	50,7812	4,6875	-17,1876	55,0781	53,516	3,5156	-14,453
33	61,7188	53,9062	57,81	70,31	-3,9063	16,4063	62,5	79,6875	0,7812	25,7813	55,4688	80,859	-6,25	26,9532
34	40,625	61,7188	46,48	60,94	5,8594	-0,7813	53,5156	56,25	12,8906	-5,4688	46,4844	49,609	5,8594	-12,109
35	56,25	70,3125	57,03	77,73	0,7812	7,4219	56,6406	82,8125	0,3906	12,5	57,0312	77,734	0,7812	7,4219
36	50	65,2344	48,44	78,13	-1,5625	12,8906	50,3906	80,8594	0,3906	15,625	56,6406	80,469	6,6406	15,2344

Tabla 4.16 - Mejora de precisión (“policy” y “value”) del conjunto 1000 respecto a Maia-1900 en los pasos de entrenamiento 100, 500 y 1000 utilizando el nuevo learning rate

Jugador	P (0)	V (0)	P (500)	V (500)	MP (500)	MV (500)	P (1000)	V (1000)	MP (1000)	MV (1000)	P (met)	V (met)	MP (met)	MV (met)
1	38,2812	73,8281	50	83,2	11,7188	9,375	45,7031	79,6875	7,4219	5,8594	41,4062	82,422	3,125	8,5938
2	36,3281	64,4531	47,66	69,53	11,3281	5,0781	44,5312	74,2188	8,2031	9,7657	39,4531	65,234	3,125	0,7813
3	48,8281	57,4219	49,61	59,38	0,7813	1,9531	46,875	62,5	-1,9531	5,0781	47,6562	68,75	-1,1719	11,3281
4	44,9219	69,5312	40,63	80,47	-4,2969	10,9376	46,875	84,7656	1,9531	15,2344	41,4062	78,906	-3,5157	9,375
5	44,9219	51,5625	50	53,91	5,0781	2,3437	55,0781	56,6406	10,1562	5,0781	50	53,906	5,0781	2,3437
6	33,2031	57,4219	43,36	85,55	10,1563	28,125	40,2344	84,375	7,0313	26,9531	44,9219	82,813	11,7188	25,3906
7	47,6562	71,875	47,27	69,53	-0,3906	-2,3438	48,8281	67,1875	1,1719	-4,6875	52,7344	63,281	5,0782	-8,5938
8	48,8281	66,4062	47,27	54,69	-1,5625	-11,719	46,0938	53,125	-2,7343	-13,2812	52,3438	50	3,5157	-16,406
9	36,3281	73,8281	44,53	75	8,2031	1,1719	43,3594	72,2656	7,0313	-1,5625	43,75	73,828	7,4219	0
10	44,5312	47,6562	61,33	59,38	16,7969	11,7188	51,5625	63,2812	7,0313	15,625	51,5625	63,281	7,0313	15,625
11	50,7812	66,4062	53,52	68,75	2,7344	2,3438	52,3438	67,9688	1,5626	1,5626	52,3438	67,969	1,5626	1,5626
12	44,1406	61,7188	48,83	62,5	4,6875	0,7812	42,9688	60,1562	-1,1718	-1,5626	42,9688	60,156	-1,1718	-1,5626
13	51,5625	43,3594	54,3	71,88	2,7344	28,5156	54,2969	67,1875	2,7344	23,8281	55,4688	66,406	3,9063	23,0468
14	45,7031	69,9219	55,86	66,8	10,1563	-3,125	51,5625	65,2344	5,8594	-4,6875	57,8125	64,844	12,1094	-5,0781
15	51,1719	62,8906	51,95	57,03	0,7812	-5,8594	51,5625	62,1094	0,3906	-0,7812	55,4688	64,453	4,2969	1,5625
16	53,9062	52,3438	51,56	51,95	-2,3437	-0,3907	59,375	58,5938	5,4688	6,25	42,1875	57,813	-11,7187	5,4687
17	50	64,8438	57,42	64,45	7,4219	-0,3907	52,7344	62,8906	2,7344	-1,9532	52,3438	64,844	2,3438	0
18	47,2656	62,1094	48,83	61,72	1,5625	-0,3906	48,8281	65,2344	1,5625	3,125	55,8594	60,547	8,5938	-1,5625
19	55,0781	53,5156	53,13	63,67	-1,9531	10,1563	53,5156	68,75	-1,5625	15,2344	44,9219	65,625	-10,1562	12,1094
20	44,9219	69,5312	52,73	69,14	7,8125	-0,3906	57,0312	60,5469	12,1093	-8,9843	55,8594	61,328	10,9375	-8,2031
21	44,1406	60,1562	48,05	58,2	3,9063	-1,9531	48,0469	54,2969	3,9063	-5,8593	46,875	53,125	2,7344	-7,0312
22	47,2656	76,1719	52,73	64,06	5,4688	-12,109	46,4844	67,5781	-0,7812	-8,5938	53,125	66,797	5,8594	-9,375
23	55,0781	57,0312	55,86	58,2	0,7813	1,1719	58,5938	60,1562	3,5157	3,125	60,5469	55,859	5,4688	-1,1718
24	51,5625	45,3125	51,17	56,25	-0,3906	10,9375	58,5938	65,625	7,0313	20,3125	52,3438	66,797	0,7813	21,4844
25	54,6875	69,9219	48,44	60,94	-6,25	-8,9844	51,9531	61,7188	-2,7344	-8,2031	51,9531	61,719	-2,7344	-8,2031

26	52,3438	61,7188	57,81	58,2	5,4687	-3,5157	53,5156	56,6406	1,1718	-5,0782	50,7812	64,453	-1,5626	2,7343
27	52,7344	54,2969	53,91	57,03	1,1718	2,7343	49,6094	62,8906	-3,125	8,5937	49,6094	62,891	-3,125	8,5937
28	49,6094	65,2344	54,3	60,55	4,6875	-4,6875	56,25	66,7969	6,6406	1,5625	56,25	67,188	6,6406	1,9531
29	57,0312	66,7969	65,23	67,97	8,2032	1,1719	52,7344	63,2812	-4,2968	-3,5157	56,6406	63,672	-0,3906	-3,125
30	53,9062	62,1094	58,59	55,08	4,6876	-7,0313	63,6719	66,7969	9,7657	4,6875	61,3281	58,984	7,4219	-3,125
31	54,2969	53,125	57,42	61,72	3,125	8,5938	62,5	63,2812	8,2031	10,1562	62,5	63,281	8,2031	10,1562
32	47,2656	61,3281	55,86	58,59	8,5938	-2,7343	50,7812	55,4688	3,5156	-5,8593	55,4688	50	8,2032	-11,328
33	60,9375	61,3281	55,47	76,17	-5,4687	14,8438	53,9062	70,3125	-7,0313	8,9844	55,8594	73,828	-5,0781	12,5
34	40,625	57,4219	43,36	55,08	2,7344	-2,3438	43,75	54,2969	3,125	-3,125	43,3594	50,781	2,7344	-6,6407
35	47,6562	71,0938	56,25	69,53	8,5938	-1,5626	55,4688	71,0938	7,8126	0	54,6875	73,438	7,0313	2,3437
36	48,8281	57,8125	50,78	78,91	1,9531	21,0937	52,7344	76,5625	3,9063	18,75	53,9062	73,828	5,0781	16,0156

Tabla 4.17 - Mejora de precisión (“policy” y “value”) del conjunto 500 respecto a Maia-1900 en los pasos de entrenamiento 100, 500 y 1000 utilizando el nuevo learning rate

Jugador	P (0)	V (0)	P (500)	V (500)	MP (500)	MV (500)	P (1000)	V (1000)	MP (1000)	MV (1000)	P (met)	V (met)	MP (met)	MV (met)
1	41,0156	69,9219	44,14	75,78	3,125	5,8593	44,1406	75	3,125	5,0781	43,3594	79,297	2,3438	9,375
2	35,1562	61,7188	44,53	61,33	9,375	-0,3907	45,3125	68,3594	10,1563	6,6406	38,6719	67,188	3,5157	5,4687
3	41,0156	60,5469	44,53	55,08	3,5156	-5,4688	45,3125	55,0781	4,2969	-5,4688	45,3125	55,078	4,2969	-5,4688
4	37,8906	63,6719	40,63	69,92	2,7344	6,25	37,5	72,6562	-0,3906	8,9843	32,8125	69,531	-5,0781	5,8593
5	42,5781	56,25	51,56	69,92	8,9844	13,6719	50,3906	68,75	7,8125	12,5	48,0469	67,969	5,4688	11,7188
6	40,2344	73,4375	48,44	76,17	8,2031	2,7344	42,5781	78,9062	2,3437	5,4687	36,7188	77,344	-3,5156	3,9063
7	45,7031	54,6875	53,91	65,63	8,2031	10,9375	55,0781	65,2344	9,375	10,5469	54,2969	72,266	8,5938	17,5781
8	44,5312	64,0625	40,23	57,81	-4,2968	-6,25	38,6719	50,3906	-5,8593	-13,6719	38,2812	52,344	-6,25	-11,719
9	37,1094	62,8906	41,02	72,27	3,9062	9,375	31,25	72,2656	-5,8594	9,375	43,3594	67,188	6,25	4,2969
10	40,625	60,1562	54,3	50,39	13,6719	-9,7656	62,1094	52,7344	21,4844	-7,4218	54,6875	52,344	14,0625	-7,8124
11	51,9531	67,9688	48,44	64,84	-3,5156	-3,125	52,3438	63,6719	0,3907	-4,2969	49,6094	69,922	-2,3437	1,9531
12	39,0625	65,2344	40,23	60,16	1,1719	-5,0782	46,4844	63,6719	7,4219	-1,5625	46,4844	63,672	7,4219	-1,5625
13	54,6875	64,4531	57,81	66,41	3,125	1,9531	48,0469	67,5781	-6,6406	3,125	48,0469	67,578	-6,6406	3,125
14	40,625	61,7188	51,95	53,91	11,3281	-7,8126	53,125	57,8125	12,5	-3,9063	49,6094	59,766	8,9844	-1,9532
15	57,4219	71,4844	53,13	67,19	-4,2969	-4,2969	50	60,9375	-7,4219	-10,5469	54,2969	56,641	-3,125	-14,844
16	46,0938	64,4531	49,61	61,72	3,5156	-2,7343	46,0938	58,2031	0	-6,25	45,7031	58,203	-0,3907	-6,25
17	43,75	69,9219	52,73	54,3	8,9844	-15,625	54,6875	53,5156	10,9375	-16,4063	50	61,719	6,25	-8,2031
18	47,6562	70,7031	49,61	63,28	1,9532	-7,4219	44,9219	69,5312	-2,7343	-1,1719	50,3906	58,984	2,7344	-11,719
19	52,3438	51,9531	46,48	55,86	-5,8594	3,9063	49,2188	62,1094	-3,125	10,1563	58,9844	56,641	6,6406	4,6875
20	47,2656	54,6875	53,13	64,06	5,8594	9,375	50,3906	59,7656	3,125	5,0781	51,9531	57,422	4,6875	2,7344
21	43,75	55,4688	39,06	53,52	-4,6875	-1,9532	49,6094	51,1719	5,8594	-4,2969	37,8906	57,422	-5,8594	1,9531
22	56,6406	64,4531	57,03	73,44	0,3906	8,9844	54,2969	60,5469	-2,3437	-3,9062	60,5469	67,969	3,9063	3,5157
23	58,5938	67,5781	56,25	62,5	-2,3438	-5,0781	57,4219	64,0625	-1,1719	-3,5156	63,2812	60,156	4,6874	-7,4219
24	53,5156	65,2344	51,17	53,91	-2,3437	-11,328	51,5625	54,6875	-1,9531	-10,5469	49,2188	57,813	-4,2968	-7,4219
25	51,1719	74,2188	45,31	64,84	-5,8594	-9,375	46,0938	60,5469	-5,0781	-13,6719	50	64,453	-1,1719	-9,7657

26	48,0469	62,5	57,81	54,69	9,7656	-7,8125	52,7344	57,8125	4,6875	-4,6875	45,7031	63,281	-2,3438	0,7812
27	52,3438	55,8594	51,56	52,73	-0,7813	-3,125	45,3125	57,4219	-7,0313	1,5625	45,3125	57,422	-7,0313	1,5625
28	51,5625	66,4062	59,38	62,5	7,8125	-3,9062	53,9062	53,5156	2,3437	-12,8906	52,7344	65,234	1,1719	-1,1718
29	56,6406	57,4219	51,95	60,94	-4,6875	3,5156	52,3438	66,7969	-4,2968	9,375	52,3438	66,797	-4,2968	9,375
30	57,4219	68,3594	59,77	69,92	2,3437	1,5625	58,5938	61,7188	1,1719	-6,6406	58,5938	61,719	1,1719	-6,6406
31	57,8125	73,8281	58,59	64,45	0,7813	-9,375	60,5469	64,0625	2,7344	-9,7656	49,2188	51,563	-8,5937	-22,266
32	50	60,5469	57,81	57,81	7,8125	-2,7344	48,4375	57,0312	-1,5625	-3,5157	51,9531	55,469	1,9531	-5,0781
33	48,8281	63,2812	58,98	69,14	10,1563	5,8594	56,25	69,1406	7,4219	5,8594	56,25	69,141	7,4219	5,8594
34	33,9844	58,9844	35,16	58,98	1,1718	0	37,8906	57,0312	3,9062	-1,9532	41,0156	61,328	7,0312	2,3437
35	51,1719	59,375	51,95	65,63	0,7812	6,25	52,3438	59,375	1,1719	0	52,3438	59,375	1,1719	0
36	48,4375	68,3594	51,56	81,64	3,125	13,2812	45,7031	78,125	-2,7344	9,7656	48,8281	70,703	0,3906	2,3437

Tabla 4.18 - Mejora de precisión (“policy” y “value”) del conjunto 250 respecto a Maia-1900 en los pasos de entrenamiento 100, 500 y 1000 utilizando el nuevo learning rate

Jugador	P (0)	V (0)	P (500)	V (500)	MP (500)	MV (500)	P (1000)	V (1000)	MP (1000)	MV (1000)	P (met)	V (met)	MP (met)	MV (met)
1	40,2344	70,3125	34,77	76,56	-5,4688	6,25	42,5781	74,2188	2,3437	3,9063	42,5781	73,828	2,3437	3,5156
2	41,7969	58,2031	33,98	59,77	-7,8125	1,5625	38,2812	62,8906	-3,5157	4,6875	35,9375	64,844	-5,8594	6,6407
3	53,5156	66,0156	37,11	55,47	-16,4062	-10,547	41,7969	58,9844	-11,7187	-7,0312	41,4062	63,672	-12,1094	-2,3437
4	40,2344	64,4531	39,06	72,66	-1,1719	8,2031	35,1562	73,8281	-5,0782	9,375	39,8438	76,563	-0,3906	12,1094
5	47,2656	52,7344	34,38	70,7	-12,8906	17,9687	42,1875	69,5312	-5,0781	16,7968	38,2812	64,063	-8,9844	11,3281
6	41,4062	69,1406	42,19	78,13	0,7813	8,9844	44,9219	75	3,5157	5,8594	31,6406	70,703	-9,7656	1,5625
7	44,1406	51,9531	43,36	67,97	-0,7812	16,0157	48,0469	68,75	3,9063	16,7969	47,6562	61,719	3,5156	9,7657
8	42,9688	40,2344	41,02	70,31	-1,9532	30,0781	41,0156	66,7969	-1,9532	26,5625	41,0156	66,797	-1,9532	26,5625
9	44,1406	60,9375	39,84	80,86	-4,2968	19,9219	32,8125	79,6875	-11,3281	18,75	32,8125	79,688	-11,3281	18,75
10	41,4062	61,3281	53,52	54,69	12,1094	-6,6406	54,2969	51,9531	12,8907	-9,375	55,0781	56,25	13,6719	-5,0781
11	45,3125	68,75	50	76,17	4,6875	7,4219	49,6094	73,8281	4,2969	5,0781	49,2188	73,438	3,9063	4,6875
12	44,5312	50,7812	39,84	58,59	-4,6874	7,8126	40,2344	58,9844	-4,2968	8,2032	34,7656	52,734	-9,7656	1,9532
13	50	59,7656	48,44	68,36	-1,5625	8,5938	53,125	64,8438	3,125	5,0782	46,0938	67,969	-3,9062	8,2032
14	36,7188	54,6875	43,75	52,73	7,0312	-1,9531	47,6562	51,1719	10,9374	-3,5156	56,25	52,344	19,5312	-2,3437
15	43,75	67,9688	43,36	54,3	-0,3906	-13,672	50,3906	59,375	6,6406	-8,5938	52,3438	58,984	8,5938	-8,9844
16	53,125	49,6094	47,27	50,39	-5,8594	0,7812	49,2188	52,7344	-3,9062	3,125	48,8281	51,172	-4,2969	1,5625
17	52,7344	57,0312	49,61	56,25	-3,125	-0,7812	53,5156	58,9844	0,7812	1,9532	51,1719	57,422	-1,5625	0,3907
18	43,3594	56,25	48,44	59,77	5,0781	3,5156	44,1406	57,8125	0,7812	1,5625	48,4375	59,766	5,0781	3,5156
19	60,9375	48,8281	51,17	52,73	-9,7656	3,9063	57,8125	46,4844	-3,125	-2,3437	51,5625	52,734	-9,375	3,9063
20	55,4688	60,9375	45,7	56,64	-9,7657	-4,2969	49,2188	57,0312	-6,25	-3,9063	46,875	60,547	-8,5938	-0,3906
21	40,2344	49,2188	40,23	48,83	0	-0,3907	39,8438	62,1094	-0,3906	12,8906	46,0938	51,172	5,8594	1,9531
22	50,7812	64,0625	52,34	55,47	1,5626	-8,5937	56,6406	58,2031	5,8594	-5,8594	54,2969	49,219	3,5157	-14,844
23	60,1562	61,3281	58,98	57,81	-1,1718	-3,5156	50	58,2031	-10,1562	-3,125	50	58,203	-10,1562	-3,125
24	49,6094	57,8125	50,39	49,22	0,7812	-8,5937	54,2969	53,5156	4,6875	-4,2969	54,2969	53,125	4,6875	-4,6875
25	54,2969	68,3594	45,31	59,77	-8,9844	-8,5938	47,6562	58,9844	-6,6407	-9,375	46,875	56,641	-7,4219	-11,719

26	50	66,0156	48,44	57,42	-1,5625	-8,5937	50,7812	54,2969	0,7812	-11,7187	50,7812	54,297	0,7812	-11,719
27	51,5625	51,9531	52,73	61,33	1,1719	9,375	45,7031	57,0312	-5,8594	5,0781	57,0312	62,109	5,4687	10,1563
28	51,5625	55,0781	51,56	50	0	-5,0781	54,6875	51,5625	3,125	-3,5156	51,1719	49,219	-0,3906	-5,8593
29	53,125	62,5	47,66	58,59	-5,4688	-3,9062	48,8281	68,75	-4,2969	6,25	49,2188	71,094	-3,9062	8,5938
30	58,2031	55,0781	56,64	64,06	-1,5625	8,9844	55,8594	59,375	-2,3437	4,2969	56,6406	57,813	-1,5625	2,7344
31	58,9844	61,7188	53,91	54,69	-5,0782	-7,0313	54,6875	57,0312	-4,2969	-4,6876	52,7344	57,422	-6,25	-4,2969
32	46,875	67,9688	48,44	60,16	1,5625	-7,8126	49,2188	59,375	2,3438	-8,5938	48,0469	60,938	1,1719	-7,0313
33	50	60,1562	58,98	67,58	8,9844	7,4219	52,3438	55,8594	2,3438	-4,2968	55,8594	63,672	5,8594	3,5157
34	42,9688	51,1719	38,28	48,44	-4,6876	-2,7344	33,9844	58,9844	-8,9844	7,8125	39,4531	45,703	-3,5157	-5,4688
35	52,3438	69,9219	49,22	64,45	-3,125	-5,4688	48,0469	62,1094	-4,2969	-7,8125	55,0781	67,188	2,7343	-2,7344
36	52,3438	57,0312	42,58	75,39	-9,7657	18,3594	45,7031	65,625	-6,6407	8,5938	49,2188	69,531	-3,125	12,5

Tabla 4.19 - Mejora de precisión (“policy” y “value”) del conjunto 100 respecto a Maia-1900 en los pasos de entrenamiento 100, 500 y 1000 utilizando el nuevo learning rate

Jugador	P (0)	V (0)	P (500)	V (500)	MP (500)	MV (500)	P (1000)	V (1000)	MP (1000)	MV (1000)	P (met)	V (met)	MP (met)	MV (met)
1	39,0625	65,2344	43,36	71,09	4,2969	5,8594	40,2344	73,0469	1,1719	7,8125	37,8906	72,656	-1,1719	7,4218
2	42,9688	55,0781	36,33	64,45	-6,6407	9,375	41,7969	64,8438	-1,1719	9,7657	41,0156	61,719	-1,9532	6,6407
3	46,0938	58,9844	42,58	66,41	-3,5157	7,4218	37,5	71,875	-8,5938	12,8906	39,0625	61,719	-7,0313	2,7344
4	41,7969	66,0156	37,11	72,66	-4,6875	6,6406	36,7188	71,4844	-5,0781	5,4688	38,2812	79,297	-3,5157	13,2813
5	37,8906	60,1562	32,03	72,66	-5,8594	12,5	35,9375	65,625	-1,9531	5,4688	32,0312	70,703	-5,8594	10,5469
6	40,625	71,4844	33,98	79,3	-6,6406	7,8125	33,9844	85,1562	-6,6406	13,6718	35,9375	75,391	-4,6875	3,9062
7	47,2656	55,0781	48,44	69,92	1,1719	14,8438	48,0469	73,0469	0,7813	17,9688	44,1406	71,484	-3,125	16,4063
8	38,2812	46,875	44,53	59,77	6,25	12,8906	39,4531	70,3125	1,1719	23,4375	39,0625	69,531	0,7813	22,6562
9	37,8906	54,2969	36,33	68,36	-1,5625	14,0625	36,3281	70,7031	-1,5625	16,4062	39,4531	74,219	1,5625	19,9219
10	50	55,8594	48,83	52,34	-1,1719	-3,5156	48,8281	51,1719	-1,1719	-4,6875	48,8281	52,344	-1,1719	-3,5156
11	44,1406	69,5312	47,27	69,92	3,125	0,3907	49,2188	73,0469	5,0782	3,5157	50,7812	71,875	6,6406	2,3438
12	37,1094	64,8438	36,33	71,88	-0,7813	7,0312	38,2812	69,1406	1,1718	4,2968	42,1875	74,219	5,0781	9,375
13	55,0781	55,0781	50,78	68,36	-4,2969	13,2813	50	65,2344	-5,0781	10,1563	50,7812	73,047	-4,2969	17,9688
14	46,875	64,8438	48,44	51,95	1,5625	-12,891	49,2188	48,4375	2,3438	-16,4063	49,2188	48,438	2,3438	-16,406
15	44,5312	55,8594	52,34	59,38	7,8126	3,5156	48,4375	54,6875	3,9063	-1,1719	46,4844	61,719	1,9532	5,8594
16	52,3438	51,1719	47,27	51,56	-5,0782	0,3906	46,0938	54,6875	-6,25	3,5156	47,2656	51,563	-5,0782	0,3906
17	51,9531	71,0938	48,44	60,94	-3,5156	-10,156	55,0781	62,8906	3,125	-8,2032	46,0938	61,719	-5,8593	-9,375
18	48,8281	67,9688	39,45	61,72	-9,375	-6,25	40,625	61,3281	-8,2031	-6,6407	40,2344	67,578	-8,5937	-0,3907
19	60,5469	58,2031	54,3	60,94	-6,25	2,7344	52,3438	58,9844	-8,2031	0,7813	54,6875	58,984	-5,8594	0,7813
20	49,6094	63,2812	43,75	62,5	-5,8594	-0,7812	42,9688	53,5156	-6,6406	-9,7656	47,6562	61,328	-1,9532	-1,9531
21	44,9219	58,2031	36,72	46,48	-8,2031	-11,719	33,5938	47,2656	-11,3281	-10,9375	39,4531	46,484	-5,4688	-11,719
22	55,0781	60,5469	55,86	60,94	0,7813	0,3906	56,6406	56,6406	1,5625	-3,9063	60,9375	60,547	5,8594	0
23	53,5156	62,8906	52,34	53,91	-1,1718	-8,9844	53,9062	51,9531	0,3906	-10,9375	52,3438	53,906	-1,1718	-8,9844
24	57,4219	53,125	50,39	52,73	-7,0313	-0,3906	44,9219	55,4688	-12,5	2,3438	47,6562	55,469	-9,7657	2,3438
25	51,9531	73,0469	48,83	59,77	-3,125	-13,281	46,875	58,5938	-5,0781	-14,4531	44,1406	52,734	-7,8125	-20,313

26	51,9531	57,0312	45,31	49,61	-6,6406	-7,4218	51,5625	54,2969	-0,3906	-2,7343	52,7344	55,078	0,7813	-1,9531
27	51,1719	60,5469	50,39	51,56	-0,7813	-8,9844	51,5625	56,6406	0,3906	-3,9063	51,5625	54,688	0,3906	-5,8594
28	-	-	-	-	-	-	-	-	-	-	-	-	-	-
29	54,6875	60,9375	52,34	60,55	-2,3437	-0,3906	52,3438	60,1562	-2,3437	-0,7813	53,5156	59,766	-1,1719	-1,1719
30	58,9844	71,875	59,77	60,55	0,7812	-11,328	53,5156	66,4062	-5,4688	-5,4688	59,7656	64,453	0,7812	-7,4219
31	54,6875	64,8438	56,64	62,5	1,9531	-2,3438	58,5938	60,9375	3,9063	-3,9063	57,4219	61,719	2,7344	-3,125
32	39,4531	59,7656	53,91	51,56	14,4531	-8,2031	50,3906	52,3438	10,9375	-7,4218	44,5312	55,469	5,0781	-4,2968
33	61,3281	60,1562	49,61	56,25	-11,7187	-3,9062	49,6094	62,1094	-11,7187	1,9532	49,2188	58,984	-12,1093	-1,1718
34	46,4844	57,8125	31,64	52,73	-14,8438	-5,0781	35,9375	46,875	-10,5469	-10,9375	31,6406	52,734	-14,8438	-5,0781
35	50,3906	48,4375	51,95	56,64	1,5625	8,2031	53,9062	55,0781	3,5156	6,6406	50,3906	53,516	0	5,0781
36	42,1875	72,2656	46,48	70,31	4,2969	-1,9531	46,875	81,6406	4,6875	9,375	46,875	77,344	4,6875	5,0782

Tabla 4.20 - Mejora de precisión (“policy” y “value”) del conjunto 50 respecto a Maia-1900 en los pasos de entrenamiento 100, 500 y 1000 utilizando el nuevo learning rate

Como se puede ver, los resultados obtenidos con este nuevo learning rate son mejores que los conseguidos en las pruebas del apartado anterior. Por lo general, a lo largo de los diferentes conjuntos se consiguen alcanzar precisiones más altas, sobre todo en aquellos con un mayor número de partidas. De hecho, si se comparan las precisiones obtenidas en el conjunto base con las logradas utilizando este learning rate en los conjuntos 1000 y 500, se puede ver que no existe una diferencia significativa entre ellas, a pesar de que estos conjuntos están formados por miles de partidas menos. Sin embargo, cuanto más se reduce el número de partidas, más se parecen las precisiones conseguidas a las alcanzadas en el apartado anterior.

Por otro lado, no parece que las precisiones logradas en los pasos en los que se obtiene el error mínimo total del entrenamiento (“met”) sean mejores que las conseguidas en el resto de pasos estudiados. Esto puede deberse a que la red no consigue generalizar bien el estilo de juego del jugador, ya que el error total es un dato que se obtiene a partir del conjunto de entrenamiento, mientras que las precisiones mostradas en estas tablas se consiguen a partir del conjunto de test.

Tampoco parece importar en gran medida el número de pasos que se utilicen para el entrenamiento. Existen jugadores que obtienen las mejores precisiones con 1000, otros con 500 y otros con los que alcanzan el mínimo error total, pero de manera general no hay un número de pasos que destaque sobre los demás.

En conclusión, estas tablas demuestran que el learning rate utilizado durante las pruebas del apartado anterior afectaba negativamente al entrenamiento de las redes neuronales. Gracias al nuevo learning rate se consiguen mejores precisiones tanto de “policy” como de “value”, a pesar de que este no se ha seleccionado de una manera rigurosa y, por tanto, probablemente no sea óptimo, las cuales son comparables con las obtenidas en el conjunto base. Sin embargo, el uso de este learning rate no parece tan determinante en los conjuntos que están formados por un número menor de partidas.

De todas formas, con esta configuración se puede decir que se cumplen los objetivos propuestos en este trabajo; el tiempo de entrenamiento es muy reducido, independientemente

de si se utilizan 500 pasos, 1000 o los pasos en los que se alcanza el “met”, y las precisiones logradas con ellos en los conjuntos 1000 y 500 son muy similares a las obtenidas en el conjunto base. Por otra parte, los resultados conseguidos con un número menor de partidas, a pesar de que difieren en mayor medida con los logrados en el conjunto base, pueden servir para entrenar la primera red del jugador a la espera de que este juegue más partidas y esta pueda mejorarse. Hay que tener en cuenta que, aunque el número de 50 partidas parezca un valor demasiado elevado para poder empezar a entrenar la primera red (sobre todo si se tiene en cuenta que en los juegos de la serie Forza y en Killer Instinct (2013) se utilizan 5 y 3 partidas respectivamente para este propósito), en el ajedrez este valor no tiene por qué ser demasiado elevado. La página de Lichess ofrece varios modos de juego estándar, de los cuales el más rápido es “Bullet 1+0” (también ofrece modos de juego personalizados en los que se pueden elegir controles de tiempo aún más reducidos). En este modo cada jugador dispone de tan solo 1 minuto para jugar la totalidad de la partida sin que se añada ningún segundo extra a su contador tras terminar su turno. Por tanto, una partida de este modo de juego como mucho dura 2 minutos, de forma que, en el peor de los casos, 50 partidas equivaldrían a 1 hora y 40 minutos de tiempo de juego. Por lo que de esta manera, en unos pocos días el jugador podría conseguir un número de partidas suficiente como para mejorar significativamente la precisión de su red.

4.6. CREACIÓN DE LIBROS DE APERTURAS

A lo largo de todo este capítulo se ha trabajado con la función de evaluación de Leela. Sin embargo, como ya se ha visto en el apartado “1.2. Ajedrez por computador en la actualidad”, a la hora de crear una IA de ajedrez no solo tiene relevancia esta función, sino que también es muy importante usar un libro de aperturas.

Como ya se sabe, la función de evaluación de un motor de ajedrez funciona correctamente durante las fases de medio juego y final. No obstante, no se puede decir lo mismo de la fase de apertura, ya que, en el caso de que la IA tenga que jugar con blancas, esta siempre realizará el mismo movimiento en el primer turno de la partida. Esto hace que las partidas pierdan dinamismo y sean más aburridas.

Para evitar este problema surgen los libros de aperturas, los cuales son bases de datos que contienen diferentes aperturas (normalmente compuestas por los 20 primeros movimientos de la partida, teniendo en cuenta los propios y los del rival) que pueden ser jugadas por la IA de ajedrez sin necesidad de utilizar la función de evaluación para evaluar el tablero. En Internet se pueden encontrar una gran cantidad de estos libros, los cuales suelen estar orientados a propósitos concretos. Por ejemplo, existen libros muy grandes con muchas aperturas, otros más pequeños para que sean más ligeros, otros que contienen aperturas ofensivas, otros defensivas, etc.

Por tanto, para evitar que las IAs creadas durante este capítulo realicen siempre el mismo primer movimiento cuando tienen que jugar como blancas, es necesario hacer uso de estos libros de aperturas. Sin embargo, hay que tener en cuenta que no tendría sentido utilizar un libro de aperturas descargado de Internet, puesto que puede contener aperturas que el jugador no realizaría nunca. Por este motivo se debe crear un libro de aperturas individualizado para cada jugador, el cual contenga únicamente las aperturas que el jugador haya utilizado previamente.

Para crear estos libros pueden utilizarse diferentes formatos, no obstante, debido a su sencillez, su popularidad y a que es un formato de código abierto, se utilizará Polyglot. Además, Polyglot es un formato de libro de aperturas binario, lo cual hace que sea más eficiente espacialmente, y que las operaciones de acceso y búsqueda sean más rápidas que en los libros de aperturas en texto [6]. La única desventaja de los libros binarios es que son más complicados de tratar para las personas, lo cual no supone un problema en este trabajo ya que solo se quiere que sean empleados por las IAs.

Por tanto, para crear estos libros de aperturas es necesario utilizar un programa que, a partir de las partidas de los jugadores, concretamente las que han jugado con blancas, obtenga las aperturas empleadas y las escriba en un fichero codificadas en el formato Polyglot. Por suerte existen varios programas que realizan esta tarea, pero por sencillez se empleará Banksia GUI, una interfaz gráfica de ajedrez libre que permite crear libros de apertura en varios formatos.

En la siguiente figura se puede observar cómo se crea un libro de aperturas a partir del fichero PGN que contiene las partidas que el jugador ha jugado como blancas. Este es un proceso muy rápido, el cual tan solo conlleva un par de segundos de ejecución, por lo que no afecta negativamente a ninguno de los objetivos de este trabajo.

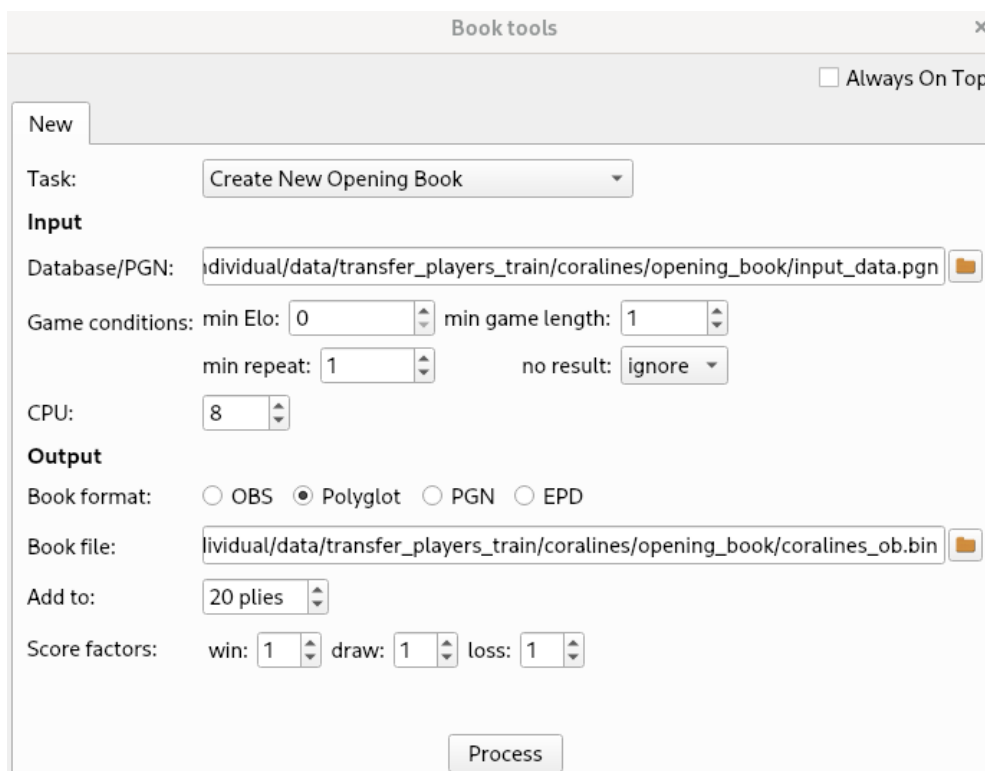


Figura 4.5 - Ventana de creación de libros de aperturas en Banksia GUI

Como se puede ver en la siguiente imagen, tras la creación del libro Banksia GUI también permite visualizar su contenido en un formato legible para una persona.

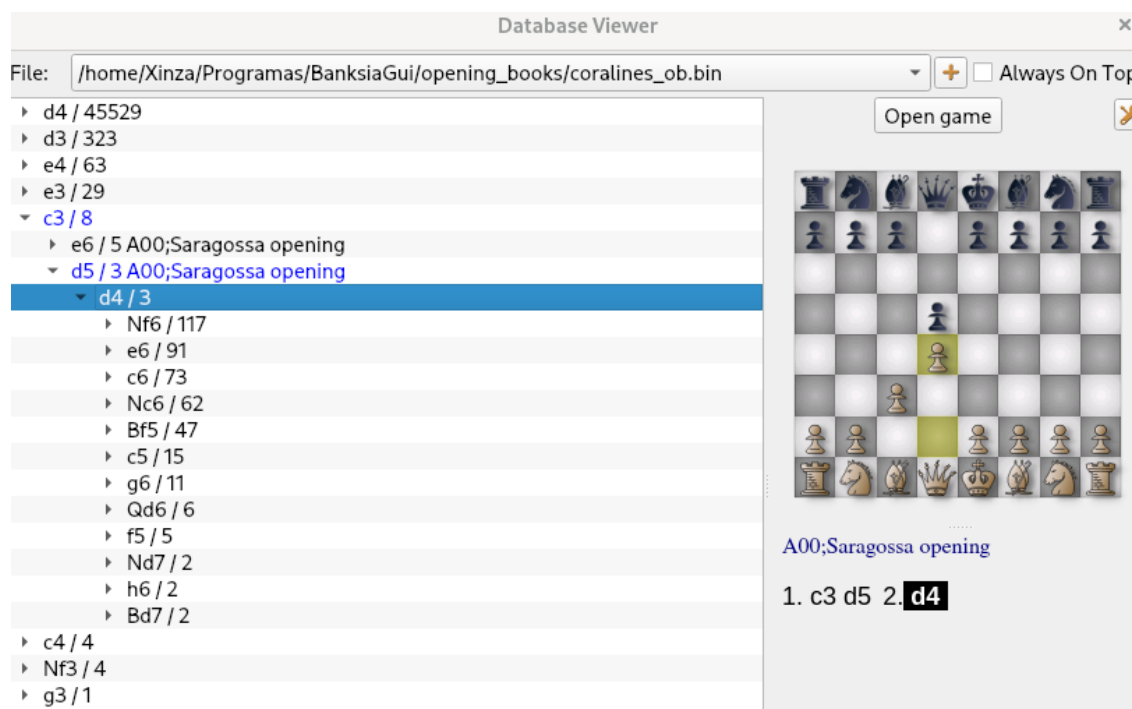


Figura 4.6 - Ventana de visualización de bases de datos en Banksia GUI

Como ya se sabe, por defecto los libros de apertura no forman parte del motor de ajedrez, sino que son archivos independientes accedidos por la interfaz gráfica, por lo que el usuario debe configurarla para que pueda utilizarlos, o indicarle que quiere que sean empleados por el motor.

5. Conclusiones y trabajos futuros

A lo largo de este documento se ha trabajado para conseguir una IA de ajedrez individualizada, que juegue con el estilo de un jugador determinado, cumpliendo a su vez tres características fundamentales: funcionar con muy pocos datos, tener la capacidad de actualizarse con nuevas partidas y que el tiempo necesario para su aprendizaje sea muy reducido.

Para conseguir estos objetivos se ha trabajado con los dos componentes que más condicionan el estilo de juego de una IA de ajedrez:

- **Función de evaluación:** a lo largo del capítulo anterior se ha estudiado y modificado la “training pipeline” de Maia, la cual es a su vez una modificación de la utilizada por Leela. Gracias a este programa se consiguen los pesos de la red neuronal que Leela utiliza como función de evaluación.
- **Libro de aperturas:** al final del capítulo anterior se ha tratado la creación de libros de aperturas individualizados, los cuales permiten a la IA emplear las mismas aperturas que utilizan los jugadores.

Aunque no sea necesario realizar ninguna modificación a la “training pipeline” empleada por Maia para conseguir el objetivo de crear una IA de ajedrez individualizada, sí que es necesario realizar algunos cambios para lograr el resto de los objetivos. En el apartado “4.5. Cambio de learning rate” puede verse la solución final empleada junto con una conclusión sobre los objetivos conseguidos, la cual puede resumirse, y complementarse, de la siguiente manera:

- **Funcionamiento con muy pocos datos:** en el apartado 4.5 (y en las gráficas del anexo) se puede ver que utilizando tan solo 500 partidas se consiguen precisiones similares a las obtenidas con el conjunto base, lo cual, teniendo en cuenta que el número de partidas más bajo que se ha utilizado en este conjunto es de 8568, es un muy buen resultado. También hay que tener en cuenta que aunque se consigan peores resultados al utilizar un número menor de partidas, la red neuronal puede ser

entrenada de todas formas, pudiendo utilizarse hasta alcanzar un número superior de partidas.

- **Actualización con nuevas partidas:** en el apartado 4.5 no se hace ninguna mención a este objetivo, ya que no es necesario realizar ninguna modificación sobre la “training pipeline” de Maia para conseguirlo. Como ya se sabe, para entrenar a las redes neuronales de este trabajo no se parte de cero, sino que se usan los pesos de la red Maia-1900 como punto de partida. Gracias a este diseño, tan solo es necesario sustituir estos pesos por los de la red que se quiera actualizar, sin necesidad de realizar ningún cambio en el código de dicho programa.
- **Tiempo de aprendizaje muy reducido:** en el apartado 4.5 (y en las gráficas del anexo) se puede ver que se consiguen buenas precisiones utilizando 1000 o incluso menos pasos de entrenamiento, lo cual, teniendo en cuenta que 1000 pasos equivalen a tres minutos de ejecución, es un tiempo aceptable. También merece la pena mencionar que la creación del libro de aperturas tan solo conlleva un par de segundos, por lo que no afecta negativamente al cumplimiento de este objetivo.

Como se puede ver, se han conseguido todos los objetivos propuestos en este trabajo. Sin embargo, aún hay mucho espacio para la mejora.

Para empezar, a pesar de que las IAs creadas pueden actualizarse con nuevas partidas, por motivos de tiempo no se pudo llevar a cabo ningún estudio para comprobar si el hecho de utilizar los pesos de una versión anterior de la red como punto de partida para el entrenamiento ofrece alguna ventaja respecto al uso de los pesos de Maia-1900. Hay que tener en cuenta que en todas las pruebas realizadas en este trabajo siempre se han entrenado las redes utilizando todas las partidas disponibles de los jugadores y partiendo de los pesos de Maia-1900. En caso de usar los pesos de las versiones anteriores como base del entrenamiento, sería necesario dividir las partidas en varios grupos, de forma que una serie de conjuntos se utilice para crear estas versiones anteriores y otro conjunto se emplee para la nueva versión. A simple vista da la impresión de que este método de actualización debería dar más importancia a las partidas más recientes respecto a las más antiguas, lo cual debería mejorar las precisiones obtenidas. No obstante, sin realizar las pruebas pertinentes es imposible determinar si se produciría dicha mejora.

Siguiendo con el tema de la actualización de partidas, también sería posible conseguir que se dé más importancia a las partidas más recientes utilizando otros medios. Por ejemplo, se podría multiplicar un determinado número de estas partidas para que así tengan una mayor influencia sobre los pesos de la red. Sin embargo, antes de poder emplear este método sería necesario realizar una gran cantidad de pruebas para determinar cuántas partidas multiplicar y el factor por el que deberían multiplicarse teniendo en cuenta el número total de partidas disponibles.

Por otra parte, los controles de tiempo utilizados en las partidas influyen directamente en el tiempo que los jugadores tienen que invertir para conseguir un número suficiente de partidas como para entrenar una IA. Como ya se ha dicho, las partidas con controles de tiempo reducidos, como los utilizados en los modos de juego “Bullet” y “Blitz”, no suponen ningún problema, puesto que el jugador puede disputar un gran número de partidas en un periodo de tiempo bastante corto. Sin embargo, si el jugador participa en partidas con controles de tiempo largos, como los utilizados en los modos clásicos (30 minutos por jugador), o directamente no utiliza ningún control de tiempo, alcanzar este número de partidas puede ser un proceso extremadamente tedioso. Si bien es cierto que la mayoría de los jugadores suele decantarse por los modos de juego rápidos, los jugadores que suelen participar en los modos de juego más largos se verán afectados por este problema.

Además, los controles de tiempo que los jugadores emplean en sus partidas no solo tienen el efecto anterior, sino que también parece que logran afectar a las predicciones de las IAs. Esta hipótesis surge tras jugar varias partidas contra algunas de las IAs entrenadas a lo largo de este trabajo, donde se pudo observar que el nivel de ELO de estas no se corresponde con la calidad de sus jugadas. Aunque es cierto que el sistema de calificación de Lichess infla el nivel de ELO de los jugadores, hay que tener en cuenta que los jugadores estudiados han disputado una gran cantidad de partidas con controles de tiempo muy reducidos, por lo que han tenido que realizar movimientos más precipitados que los que harían si tuvieran más tiempo para pensarlos. Por poner un ejemplo, el jugador 21 ha disputado 77562 partidas, de las cuales 48026 han sido jugadas con el control de tiempo “0+1”, lo que significa que los jugadores empiezan la partida con un contador a 0, al cual pueden sumarle un segundo después de cada movimiento.

Teniendo en cuenta la importancia que tiene el control de tiempo en las partidas, en el futuro sería interesante poder darle la relevancia que se merece a la hora de entrenar las IAs, de forma que puedan ajustarse al estilo de juego que utiliza el jugador con cada control de tiempo. De hecho, una solución mejor sería tener en cuenta el tiempo disponible que tiene el jugador al realizar cada jugada, de forma que la calidad de sus movimientos dependa de este dato. Sin embargo, los archivos PGN no están diseñados para proporcionar esta información, por lo que resultaría imposible llevar a cabo esta tarea.

Bibliografía

- [1] ALLIS, V. *Searching for Solutions in Games and Artificial Intelligence* [en línea]. Tesis doctoral. Universidad de Limburg, Maastricht, Países bajos, 1994. [fecha de consulta: 4 diciembre 2022]. ISBN: 90-900748-8-0. Disponible en: <https://project.dke.maastrichtuniversity.nl/games/files/phd/SearchingForSolutions.pdf>
- [2] Colaboradores de Chess Programming Wiki. GUI [en línea]. En *Chess Programming Wiki*, 2022. [fecha de consulta: 4 septiembre 2023]. Disponible en: <https://www.chessprogramming.org/GUI>
- [3] Colaboradores de Chess Programming Wiki. Getting Started [en línea]. En *Chess Programming Wiki*, 2018. [fecha de consulta: 13 marzo 2024]. Disponible en: https://www.chessprogramming.org/Getting_Started
- [4] Colaboradores de Chess Programming Wiki. Evaluation [en línea]. En *Chess Programming Wiki*, 2021. [fecha de consulta: 6 septiembre 2023]. Disponible en: <https://www.chessprogramming.org/Evaluation>
- [5] Colaboradores de Chess Programming Wiki. Search [en línea]. En *Chess Programming Wiki*, 2023. [fecha de consulta: 6 septiembre 2023]. Disponible en: <https://www.chessprogramming.org/Search>
- [6] Colaboradores de Chess Programming Wiki. Opening Book [en línea]. En *Chess Programming Wiki*, 2021. [fecha de consulta: 21 mayo 2024]. Disponible en: https://www.chessprogramming.org/Opening_Book
- [7] Colaboradores de Chess Programming Wiki. Endgame tablebases [en línea]. En *Chess Programming Wiki*, 2022. [fecha de consulta: 6 septiembre 2023]. Disponible en: https://www.chessprogramming.org/Endgame_Tablebases
- [8] MEYER-KAHLEN, S. *UCI protocol* [en línea] [fecha de consulta: 7 septiembre 2023]. Disponible en: <https://wbec-ridderkerk.nl/html/UCIProtocol.html>
- [9] THOMPSON, T. How Forza's Drivatar Actually Works [en línea]. En *Game Developer*, 2021. [fecha de consulta: 24 febrero 2023]. Disponible en: <https://www.gamedeveloper.com/design/how-forza-s-drivatar-actually-works>

- [10] THOMPSON, T. The Killer Groove: The Shadow AI of Killer Instinct [en línea]. En *Game Developer*, 2017 [fecha de consulta: 26 febrero 2023]. Disponible en: <https://www.gamedeveloper.com/programming/the-killer-groove-the-shadow-ai-of-killer-instinct>
- [11] Colaboradores de Chess Programming Wiki. Leela Chess Zero [en línea]. En *Chess Programming Wiki*, 2023. [fecha de consulta: 9 abril 2024]. Disponible en: https://www.chessprogramming.org/Leela_Chess_Zero
- [12] MCILROY-YOUNG, R., SEN, S., KLEINBERG, J. y ANDERSON, A. *Aligning Superhuman AI with Human Behavior: Chess as a Model System* [en línea] [fecha de consulta: 17 febrero 2024]. Disponible en: <https://arxiv.org/pdf/2006.01855.pdf>
- [13] MCILROY-YOUNG, R., WANG, R., SEN, S., KLEINBERG, J. y ANDERSON, A. *Learning Models of Individual Behavior in Chess* [en línea] [fecha de consulta: 17 febrero 2024]. Disponible en: <https://arxiv.org/pdf/2008.10086.pdf>
- [14] EDWARDS, S. y otros. *Standard: Portable Game Notation Specification and Implementation Guide* [en línea] [fecha de consulta: 30 octubre 2023]. Disponible en: <http://www.saremba.de/chessgml/standards/pgn/pgn-complete.htm>
- [15] Colaboradores de Leela Chess Zero. Technical Explanation of Leela Chess Zero [en línea]. En *Leela Chess Zero*, 2021. [fecha de consulta: 10 mayo 2024]. Disponible en: <https://lczero.org/dev/wiki/technical-explanation-of-leela-chess-zero/>
- [16] Colaboradores de Leela Chess Zero. What is Lc0? (for non programmers) [en línea]. En *Leela Chess Zero*, 2020. [fecha de consulta: 10 mayo 2024]. Disponible en: <https://lczero.org/dev/wiki/what-is-lc0-for-non-programmers/>
- [17] Colaboradores de Leela Chess Zero. Best Nets for Lc0 [en línea]. En *Github*, 2024. [fecha de consulta: 10 mayo 2024]. Disponible en: <https://github.com/LeelaChessZero/lc0/wiki/Best-Nets-for-Lc0>
- [18] Barnes, D. *pgn-extract: A Portable Game Notation (PGN) Manipulator for Chess Games* [en línea] [fecha de consulta 18 febrero 2024]. Disponible en: <https://www.cs.kent.ac.uk/people/staff/djb/pgn-extract>
- [19] Uranga, D. y otros. trainingdata-tool [en línea]. En *Github*, 2022. [fecha de consulta 18 febrero 2024]. Disponible en: <https://github.com/DanielUranga/trainingdata-tool>
- [20] *Banksia GUI* [en línea] [fecha de consulta: 18 febrero 2024]. Disponible en: <https://banksiagui.com/>