Redes neuronales y Algoritmos Genéticos en la Era de los Videojuegos

Bartolucci Gino, Joaquín Bates, Francisco Mendiburu

Universidad Tecnológica Nacional Facultad Regional Rosario, Cátedra de Algoritmos Genéticos

Abstract

El presente trabajo se origina en la intersección de dos campos científicos altamente relevantes en la actualidad: el aprendizaje automático y los algoritmos genéticos.

El presente estudio se centra en el juego arcade "Breakout", donde los agentes controlan una paleta para lograr rebotar una pelota y eliminar una pared compuesta de bloques dispuestos en la parte superior de la pantalla. Aunque este juego puede parecer simple, presenta un entorno interactivo y dinámico con desafíos significativos debido a su naturaleza impredecible y rápida. El objetivo principal de este trabajo es investigar cómo la combinación de algoritmos genéticos y técnicas de aprendizaje automático, específicamente NeuroEvolution of Augmenting Topologies (NEAT), puede llevar a la creación de agentes que superen el rendimiento humano en el juego.

Palabras Clave

Algoritmos genéticos, crossover, redes neuronales, topologias, deep learning.

Introducción

El problema a resolver radica en desarrollar estrategias automatizadas que permitan a los agentes aprender y adaptarse en tiempo real para maximizar su puntuación en el juego Breakout. Esto implica la identificación de funciones de aptitud adecuadas, la representación eficiente del espacio de búsqueda y la adaptación de operadores genéticos para mantener la diversidad de soluciones. Además, se explorará cómo las redes neuronales, optimizadas mediante NEAT, pueden mejorar la toma de decisiones de los agentes en un entorno de juego interactivo y desafiante.

La organización del documento se compone de varias secciones que abordan la fundamentación teórica, la aplicación de los enfoques propuestos, los resultados obtenidos y su análisis.

Marco teórico

Introducción al Aprendizaje Automático y Algoritmos Genéticos

El aprendizaje automático, definido como una rama de la inteligencia artificial, permite a los sistemas computacionales aprender patrones y tomar decisiones a partir de datos, mejorando su rendimiento con la experiencia (Mitchell, T. M.)[1]. Los algoritmos genéticos, por otro lado, son una técnica de optimización inspirada en la evolución biológica, que utiliza operadores genéticos como selección, mutación y crossover para mejorar soluciones candidatas en un espacio de búsqueda (Holland, J. H.)[2].

Las redes neuronales, un componente clave del aprendizaje automático, son modelos matemáticos inspirados en la estructura y función del cerebro humano. Consisten en nodos interconectados, llamados neuronas, organizados en capas y conectados mediante ponderaciones que ajustan el impacto de cada neurona en la salida del modelo.

Redes Neuronales en el Contexto del Problema

Dentro del campo del aprendizaje automático, las redes neuronales han emergido como una herramienta poderosa para abordar problemas complejos y no lineales, como la toma de decisiones en videojuegos (LeCun, Y., et al.)[3]. En el contexto de la investigación sobre el entrenamiento de agentes inteligentes en videojuegos arcade, las redes neuronales ofrecen la capacidad de aprender representaciones y estrategias adaptativas directamente de los datos del juego. La evolución de la topología de las redes neuronales a través de algoritmos genéticos, como NEAT, permite la adaptación de la arquitectura y el rendimiento del agente a lo largo del tiempo, mejorando su capacidad

para enfrentar los desafíos cambiantes del juego.

Topología en una Red Neuronal

La topología en una red neuronal se refiere a la estructura y disposición de las neuronas y conexiones en capas dentro del modelo (Goodfellow, I., et al.)[4]. Es la configuración espacial que determina cómo las neuronas están conectadas entre sí, influyendo en la capacidad de la red para aprender y generalizar patrones complejos (Nielsen)[5]. Una topología puede ser profunda, con varias capas interconectadas, o más superficial con menos capas, y puede incluir conexiones recurrentes o saltos directos entre capas.

La elección de la topología es esencial en el diseño de una red neuronal, ya que determina su capacidad para capturar relaciones entre los datos de entrada y producir salidas precisas y significativas.

Elección de NEAT en el Proyecto:

La elección de utilizar NEAT (NeuroEvolution of Augmenting Topologies) se fundamenta en su potencial para abordar problemas de optimización en entornos complejos y dinámicos como los videojuegos (Stanley & Miikkulainen)[6].

La combinación de algoritmos genéticos y redes neuronales a través de NEAT permite la evolución de la topología, adaptación de la arquitectura y el rendimiento del agente a lo largo del proceso de evolución. Su adopción en el entrenamiento de agentes en videojuegos donde las estrategias pueden ser altamente variables presenta una oportunidad prometedora para resolver los desafíos de optimización y toma de decisiones.

Enfoque del proyecto

El enfoque se basa en la idea de que un agente aprende a tomar decisiones óptimas al interactuar con su entorno y recibir recompensas por sus acciones (Sutton & Barto)[7]. En el contexto de los videojuegos, los agentes aprenden a través de ensayo y error, ajustando sus acciones para maximizar las recompensas acumuladas a lo largo del tiempo. A medida que los agentes exploran y toman decisiones, ajustan sus políticas de acción para mejorar su desempeño en el juego.

NEAT permite la evolución de estrategias de juego a través de la adaptación tanto de la topología como de los pesos de las redes neuronales que guían las acciones del agente. Esto proporciona una solución eficaz para entrenar agentes capaces de anticipar la trayectoria de la pelota, optimizar la posición de la paleta y lograr puntuaciones sobresalientes.

La aplicación de algoritmos genéticos y redes neuronales mediante NEAT en el juego Breakout puede llevar a la generación de agentes altamente competentes, capaces de superar el rendimiento humano. Este enfoque ha explorado la adaptabilidad de las topologías neuronales y la optimización de políticas de acción.

Concreción del modelo

Descripción de la Implementación Técnica

El stack tecnológico seleccionado comprende Python como lenguaje de programación principal, junto con la biblioteca NEAT para entrenar el modelo de IA y la biblioteca PYGAME para crear el entorno de juego Breakout. Python proporciona una plataforma versátil y dinámica para el desarrollo de IA, mientras que la biblioteca NEAT ofrece un marco robusto para evolucionar redes neuronales y optimizar su estructura. La biblioteca PYGAME, por su parte, nos permite crear una interfaz de juego para poder acceder de manera sencilla a la información del entorno esencial para probar nuestro modelo y validar el rendimiento del agente de IA.

Entorno del juego Breakout

El mismo consistirá en una paleta que deberá moverse en el eje x para hacer rebotar una pelota sobre sí y así destruir todos los bloques, 36 en total, en la parte superior de la pantalla Figura 1. Al destruir los 36 bloques se considerará que el agente ha ganado el juego, por el contrario si la paleta no alcanza la pelota el agente ha

perdido el juego.
Se considerará, en caso de ganar, cuantas veces tuvo que rebotar la pelota en la paleta, teniendo en cuenta que a menor cantidad mejor será el desempeño.

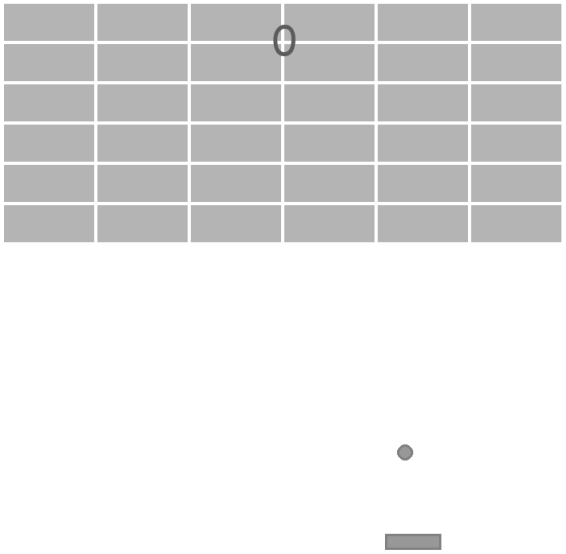

Figura 1. Juego Breakout

Estructura general del entrenamiento del agente

Se genera una población inicial aleatoria de 50 genomas, cada genoma contiene la información necesaria para crear una red neuronal, teniendo solo en común la capa de entrada y la capa de salida, esto se detalla más adelante.
Posteriormente de generar la población inicial, se inicia la ejecución del juego utilizando individualmente cada una de las redes neuronales asociadas a los respectivos genomas. Al finalizar la ejecución de juego con cada red, se procede a evaluar su rendimiento asignando un valor de aptitud (fitness[1]). En base a esto se seleccionarán los mejores y se generará un crossover[2], se los somete a una posible mutación[3] aleatoria tanto de su topología como del peso y sesgo correspondiente a alguna de sus neuronas. Finalmente se habrá obtenido una nueva población, es decir una nueva generación de genomas, el proceso se repetirá hasta obtener una red neuronal capaz de ganar el juego en diferentes contextos y con la menor cantidad posible de rebotes de la pelota contra la paleta.

Estructura común de las redes neuronales

Los genomas que representan la estructura de las redes neuronales comparten dos capas fundamentales, la capa de entrada y la capa de salida. La primera está compuesta por tres neuronas, cada una de las cuales cumple una función específica en tiempo real. Estas neuronas capturan información crítica para el agente. En primer lugar, una neurona refleja la posición actual de la paleta en el eje horizontal (x). La segunda neurona codifica la distancia actual entre la pelota y la paleta, lo que proporciona al agente una medida de la proximidad entre estos elementos esenciales del juego. La tercera neurona en la capa de entrada representa la posición actual de la pelota en el eje horizontal (x).
La capa de salida, por otro lado, está compuesta por tres neuronas que representan las decisiones que el agente debe tomar en tiempo real para interactuar eficazmente con el entorno del juego. La primera neurona de la capa de salida tiene como objetivo principal mantener la posición actual de la paleta, lo que permite al agente tomar la decisión de no realizar un desplazamiento. La segunda neurona de salida codifica la acción de desplazarse hacia la izquierda, mientras que la tercera neurona refleja la acción de desplazarse hacia la derecha. Esta estructura puede observarse en la Figura 2.

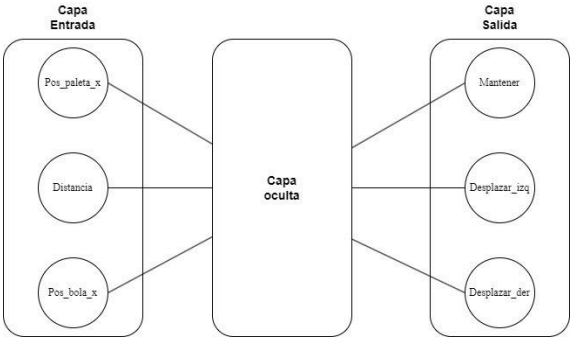

Figura 2. Estructura de la red neuronal

[1]Fitness: Es un tipo de función objetiva que se utiliza para resumir lo cerca que está una determinada solución de alcanzar los objetivos establecidos.
[2]Crossover: operación que combina información genética de dos padres para generar descendientes con la esperanza de explorar nuevas soluciones en el espacio de búsqueda.
[3] Mutación: Es una operación cuyo objetivo es generar nueva información dentro de la población de soluciones para obtener una mejor exploración del espacio de búsqueda.

En cuanto a las funciones de activación, se utilizó la función "ReLU" (Rectified Linear Unit) como función de activación predeterminada en todas las neuronas de las capas ocultas y de salida. La función "ReLU" se caracteriza por su capacidad para modelar relaciones no lineales y su eficiencia computacional.

La elección de esta arquitectura de red neuronal se basó en la capacidad de capturar eficazmente la información crítica del entorno del juego y emitir decisiones adecuadas en tiempo real. Además, esta estructura relativamente simple se complementó con el proceso de evolución de topologías proporcionado por NEAT.

Operadores genéticos

En el enfoque de este estudio, se implementaron operadores genéticos como parte del algoritmo NEAT desempeñaron un papel esencial en la búsqueda y optimización de soluciones eficaces.

La mutación se aplicó para introducir variabilidad en las redes neuronales de los agentes. Todas las neuronas de la red podrían sufrir mutaciones en su función de activación en cada generación. La función de activación es fundamental, ya que determina cómo una neurona procesa su entrada. La mutación de esta función permitió a los agentes explorar diferentes estrategias de toma de decisiones.

Mediante la mutación también se ajustaron los sesgos que afectan la salida de una neurona para adaptar la respuesta de la red a las demandas cambiantes del juego.

Los pesos de las conexiones entre neuronas influyen en la fuerza de influencia de una neurona sobre otra. La mutación de los pesos permitió a las redes neuronales adaptarse a las relaciones cambiantes entre los datos de entrada y las decisiones de salida.

Se empleó el cruce o crossover para combinar información genética de dos redes neuronales padres y crear descendencia con características heredadas de ambos. En el contexto, NEAT determina la compatibilidad entre dos redes neuronales para el cruce. Una compatibilidad alta indicaba que dos redes eran más similares y, por lo tanto, más propensas a cruzarse.

Se aplicó la estrategia de elitismo para asegurar que los cuatro individuos con el mejor rendimiento en cada generación se mantuvieran sin cambios en la siguiente generación. Los individuos élites no experimentan mutaciones ni cruces, lo que permite acelerar el proceso de búsqueda al conservar sus características sobresalientes y transmitirlas intactas a las siguientes generaciones.

Cálculo de la Aptitud del Agente

Este procedimiento fue fundamental para determinar qué tan exitoso había sido el desempeño del agente en el juego Breakout y, en última instancia, para guiar el proceso de evolución de las redes neuronales.

Tras cada iteración de juego, se evaluó la aptitud de un agente utilizando el código implementado el la Figura 3.

```
def calculate_fitness(self, genome, game_info):
    if game_info.points == self.game.rows*self.game.cols:
        genome.fitness += 50 - game_info.paddle_hits/10
    else: genome.fitness += game_info.points
```

Figura 3. Cálculo del fitness.

El cálculo de la aptitud se realizó en base a dos condiciones clave, Condición de Victoria y Condición de Puntuación Normal.

Condición de Victoria: Si el agente logra eliminar todos los bloques del juego, se asigna una puntuación de aptitud de 50 puntos que premia su éxito y se resta una cantidad proporcional a los golpes innecesarios de la paleta.

Condición de Puntuación Normal: Si el agente no lograba eliminar todos los bloques, su puntuación de aptitud se basaba simplemente en la cantidad de puntos que había acumulado durante su partida.

Este proceso de cálculo de aptitud proporcionó una forma objetiva de medir el éxito del agente en el juego, recompensando la finalización exitosa del nivel y penalizando los golpes innecesarios de la paleta. La aptitud calculada se utilizó como base para la selección y evolución de las redes neuronales a través del algoritmo

NEAT, contribuyendo así al aprendizaje y mejora continua de los agentes.

Resultados y Rendimiento

A lo largo de las 35 generaciones, se observa una tendencia constante de mejora en el fitness de los agentes. Ver Tabla 1.

El mejor fitness alcanzado aumentó progresivamente desde un valor inicial de 2 en la primera generación hasta un punto máximo de 48 en la generación 31. Este aumento en el fitness sugiere que los agentes aprendieron a desempeñarse mejor a medida que avanzaba el proceso de entrenamiento.

Además el fitness promedio también muestra una mejora general a lo largo del tiempo. Comenzando en aproximadamente 1.36 en la primera generación, el fitness promedio aumentó de manera constante y alcanzó su punto máximo de alrededor de 14.426 en la generación 34 respaldando la eficacia del enfoque de combinación de algoritmos genéticos y redes neuronales para mejorar el desempeño de los agentes en un entorno interactivo mediante NEAT .

Generacion	Mejor Fitness	Fitness Promedio
1	2	1,36
2	7	1,56
3	25	1,94
4	21	1,98
5	22	2,06
6	46,4	3,108
7	29	3,74
8	46,9	6,17
9	46,9	4,486
10	46,9	6,198
11	46,8	6,83
12	47,3	7,342
13	47,2	8,978
14	46,9	8,554
15	47,7	10,9
16	47,7	9,068
17	47,2	7,982
18	47,2	10,474
19	47,4	10,61
20	47,3	8,542
21	47	9,016
22	47,5	10,894
23	46,9	11,78
24	47,3	11,458
25	48	14,058
26	47,3	11,976
27	47,3	12,606
28	47,4	12,192
29	47,4	15,124
30	47,1	12,078
31	48	14,408
32	47,3	14,034
33	47,2	13,616
34	47,7	14,426
35	47,2	11,998

Tabla 1. Desempeño por generaciones.

Problemática: Impacto de la Variabilidad del Juego en el Desempeño de Agentes con Fitness Elevado

La variabilidad del juego puede actuar como un obstáculo para los agentes con fitness elevado. Un agente que ha perfeccionado una estrategia altamente efectiva en un conjunto específico de condiciones de juego puede enfrentar dificultades cuando se encuentra con situaciones inesperadas o variantes. Esto nos lleva a cuestionar si un alto nivel de fitness en un agente es realmente un indicativo de su capacidad para adaptarse y mantener un rendimiento sólido en todas las circunstancias.

Al someter a prueba a una red neuronal con un fitness muy alto fuera del entorno de entrenamiento, no siempre logra ganar el juego. Al emplear el genoma destacado como el más apto de la última generación en una partida en tiempo real de "Breakout", se evalúa su desempeño en un caso real.

Si bien, en general, el desempeño suele ser satisfactorio, logrando incluso ganar el juego en un número reducido de movimientos, se ha notado que ocasionalmente puede resultar en la derrota. De este fenómeno entre genomas cuyos fitness nos indican que han tenido éxito, ganando el juego, surge el siguiente interrogante: ¿es el genoma con un fitness más alto necesariamente el que mostrará un mejor desempeño en una amplia variedad de situaciones, o su fitness se debe

únicamente a la idoneidad particular de su proceso de entrenamiento?"

Evaluación del Interrogante a través de Pruebas Iterativas

Con el fin de someter a examen el interrogante planteado, se llevó a cabo un conjunto de pruebas sistemáticas en las que se puso a jugar el juego un total de 100 veces en condiciones reales y no de entrenamiento utilizando el genoma que obtuvo el mejor rendimiento. Este genoma fue seleccionado después de ser entrenado a lo largo de 5 generaciones. Adicionalmente, se realizaron pruebas similares utilizando el genoma ganador después de 10, 25 y 30 generaciones de entrenamiento.

Procedimiento Experimental

Durante cada una de las 100 repeticiones del juego, se registraron y analizaron los resultados obtenidos. Se evaluó si el genoma con el mejor rendimiento durante el entrenamiento mantuvo su capacidad para ganar el juego en diversas situaciones, o si su éxito se vio influenciado por la duración específica del entrenamiento.

Resultados

Estos resultados, ver Tabla 2, indican el porcentaje de victorias obtenido por el genoma seleccionado después de diferentes períodos de entrenamiento, así como el mejor valor de fitness alcanzado en cada caso.

Muestran un aumento significativo en el rendimiento a medida que aumenta la duración del entrenamiento, evidenciado en la generación 30, donde el genoma logra un porcentaje de victorias del 97% y un fitness de 47.1 mientras que la generación 25 con un fitness de 48 tiene un porcentaje de victorias del 77%. A pesar de tener un mayor fitness su desempeño en una amplia gama de casos reales es menor.

Conclusiones Preliminares

Si bien un genoma con un alto fitness puede tener éxito en el juego después de un entrenamiento prolongado, la capacidad de un genoma para adaptarse y tener éxito en diferentes contextos de juego parece estar relacionada con la cantidad de generaciones de entrenamiento a la que se somete y no tanto con su fitness.

Estos resultados subrayan la importancia de un entrenamiento extenso y diversificado en la formación de agentes inteligentes capaces de enfrentar desafíos variables en entornos de juego.

Es importante destacar que estos hallazgos son preliminares y pueden servir como base para investigaciones adicionales en esta área.

Generacion	Porcentaje de victorias	Mejor fitness
5	54%	22
10	62%	46,9
25	77%	48
30	97%	47,1

Tabla 2. Porcentajes de victorias en 100 juegos reales.

Trabajos Relacionados

Investigaciones que han utilizado NEAT para entrenar agentes en juegos son: Super Mario (Togelius et al., 2009)[8], Flappy Bird (Xie et al., 2017)[9]

Conclusión y Trabajos Futuros

En este estudio, se ha explorado con éxito la utilización de algoritmos genéticos y la neuroevolución mediante el algoritmo NEAT. Se han obtenido resultados prometedores que indican que esta combinación de enfoques puede llevar a mejoras significativas en el rendimiento de los agentes en entornos de juego dinámicos y complejos.

Se ha demostrado que NEAT permite la evolución de topologías neuronales, lo que significa que las redes neuronales de los agentes pueden evolucionar adaptándose y cambiando su estructura continuamente a través de la interacción con su entorno para enfrentar el desafío de jugar a Breakout de manera más efectiva.

El estudio también ha destacado la importancia de la mutación y el crossover como operadores genéticos en la búsqueda y optimización de soluciones eficaces. La mutación permite la introducción de variabilidad en las redes neuronales, mientras que el crossover permite la

exploración de combinaciones genéticas que pueden llevar a mejoras en el rendimiento.

Limitaciones y Áreas de Mejora

A pesar de los resultados alentadores, este trabajo presenta algunas limitaciones que podrían abordarse en futuras investigaciones. Algunas de estas limitaciones incluyen:

Rendimiento Humano: Aunque los agentes han demostrado un rendimiento prometedor, aún no se ha alcanzado un nivel de rendimiento que supere consistentemente al rendimiento humano en el juego de Breakout. En futuras investigaciones, se podría trabajar en la mejora de las estrategias de entrenamiento y en la adaptación de la arquitectura de las redes neuronales para lograr un rendimiento superhumano.

Exploración de Hiperparámetros: Este estudio ha utilizado un conjunto específico de hiperparámetros para los operadores genéticos y la configuración de NEAT. Sería beneficioso explorar una amplia gama de hiperparámetros para determinar cómo afectan al rendimiento de los agentes y si existe una configuración óptima.

Transferencia de Aprendizaje: Investigar la capacidad de transferir el conocimiento adquirido por los agentes en el juego de Breakout a otros juegos o dominios sería una dirección interesante para futuras investigaciones. Esto podría involucrar el uso de técnicas de aprendizaje por refuerzo transferido.

Código del proyecto

El código fuente del proyecto está disponible en el repositorio público de GitHub en el siguiente enlace: https://github.com/JoaquinBetes/ProyectoAlgoritmosGeneticos

También pueden ver una demostración en el siguiente enlace: https://www.youtube.com/watch?v=0kaCRvTS7pk&t=1s

Agradecimientos

Agradecimiento a la profesora Daniela Díaz y al profesor Víctor Lombardo por la orientación y apoyo en la realización de este trabajo.

También reconocer a la Universidad Tecnológica Nacional (UTN) por proporcionar el entorno académico propicio para llevar a cabo esta investigación. La UTN FRRo nos ha brindado los recursos y el ambiente necesario para fomentar el aprendizaje y la investigación, permitiendo así el desarrollo de este trabajo.

Referencias.

[1] Mitchell, T. M. (1997). Machine Learning. McGraw-Hill.

[2] Holland, J. H. (1975). Adaptation in natural and artificial systems. University of Michigan Press.

[3] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436-444.

[4] Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT press.

[5] Nielsen, M. A. (2015). Neural networks and deep learning: A textbook. Determination Press.

[6]Stanley, K. O., & Miikkulainen, R. (2002). Evolving neural networks through augmenting topologies. Evolutionary computation, 10(2), 99-127.

[7]Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction. MIT press.

[8]Togelius, J., Yannakakis, G. N., & Stanley, K. O. (2009). Preface to the special issue on evolutionary and player-adaptive games. IEEE Transactions on Computational Intelligence and AI in Games, 1(1), 1-2.

[9] Xie, H., & Zhong, L. (2017). Playing flappy bird with NEAT. In Proceedings of the 2017 IEEE Symposium Series on Computational Intelligence (SSCI) (pp. 1-6).

Datos de Contacto:

Bartolucci Gino - ginobartolucci00@gmail.com, Joaquín Bates - betesjoaquin@gmail.com, Francisco Mendiburu - franmendi.fm@gmail.com. Universidad Tecnológica Nacional Facultad Regional Rosario. S2000BQB Rosario, Santa Fe, Argentina.