ISSN 1870-4069

Esquema de aprendizaje híbrido de agentes colaborativos en videojuegos MOBA

José A. Torres León, Marco A. Moreno Armendáriz,
Francisco Hiram Calvo Castro

Instituto Politécnico Nacional,
Centro de Investigación en Computación,
México

{jtorresl2019,mam_armendariz,hcalvo}@cic.ipn.mx

Resumen. En este artículo se presenta un esquema de aprendizaje híbrido, que busca dividir el problema de aprendizaje para un equipo de agentes jugadores de videojuegos del género Multiplayer Online Battle Arena (MOBA). Este esquema se basa en un análisis por módulos de la tarea colaborativa, la cual se descompone en situaciones de juego. Hacer el análisis a nivel de situación de juego permite dividir el problema de aprender un comportamiento colaborativo del equipo de agentes en distintas sub-tareas de aprendizaje, donde la solución a cada una de ellas requiere de un enfoque específico, ya sea supervisado, no supervisado o reforzado, que se adecue mejor a la estructura de los datos en dicha sub-tarea. Este nuevo esquema de aprendizaje híbrido debería de sentar las bases para entrenar un equipo de agentes jugadores de videojuegos MOBA capaces de resolver una misma situación de juego para múltiples videojuegos del género.

Palabras clave: Inteligencia artificial, aprendizaje automático, agentes inteligentes, tareas colaborativas.

Hybrid Learning Scheme of Collaborative Agents in MOBA Videogames

Abstract. In this paper a hybrid learning scheme is presented, which aims to divide the learning problem for a video games player agents team of the Multiplayer Online Battle Arena (MOBA) genre. This schema is based in a per module analysis of the collaborative task, which decomposes into game situations. By doing this analysis at game situation level, it is possible to divide the problem of learning a collaborative behavior of the agents team into different learning sub-tasks, where the solution to each one of them requires a specific approach, whether it's supervised, unsupervised or reinforced, which adequates better to the structure of the data of that sub-task. This new hybrid learning scheme should lay the foundation to train a team of MOBA video games player agents team capable of solving a game situation for multiple video games of the genre.

Keywords: Artificial intelligence, machine learning, intelligent agents, collaborative tasks.

1. Introducción

Actualmente, uno de los problemas abiertos de la inteligencia artificial (IA) son las tareas colaborativas, las cuales requieren de un alto grado de inteligencia, tal como lo mencionan los autores de [1] . Para estos problemas, un equipo de agentes inteligentes necesita colaborar para encontrar la solución a una tarea, coordinando sus acciones y haciendo predicciones a largo plazo. Como caso de estudio, se seleccionó el género de videojuegos Multiplayer Online Battle Arena (MOBA), que, como se explica en [5], presenta desafíos colaborativos interesantes, como la limitación de los agentes a disponer de información incompleta del juego, planear estrategias, entre otras.

La meta global de estos videojuegos es destruir la base enemiga. Para lograr esa meta global, el equipo debe coordinar sus acciones para resolver una variedad de sub-tareas, para alcanzar metas individuales que incrementarán la probabilidad de una victoria. La estrategia involucrada en estos videojuegos es dirigida por las particularidades de las sub-tareas, en otras palabras, el equipo que tiene un buen desempeño en cada tarea, tendrá una mayor probabilidad de ganar.

En este caso de estudio, dichas sub-tareas se conocen como "situaciones de juego". Los jugadores humanos detectan estas situaciones y se adaptan a ellas de manera natural, fluyendo con el ritmo del juego. Sin embargo, aún se investiga para lograr que equipos basados en algoritmos de IA logren un desempeño similar. En este trabajo, se presenta un esquema de aprendizaje automático (Machine Learning, ML) híbrido enfocado en aprender cómo lidiar con el ajuste dinámico a las situaciones para resolver las tareas colaborativas complejas presentes en los videojuegos del género MOBA.

La mayor parte de las propuestas existentes en el estado del arte que proponen equipos de jugadores artificiales de videojuegos MOBA, que se presenta en la sección 2, se basan principalmente en modelos de aprendizaje por refuerzo profundo para resolver las predicciones a largo plazo y el modelado de las sub-tareas, lo cual significa que no se hace un modelado explícito de estos fenómenos, sólo se infieren por las arquitecturas profundas reforzadas.

Por ello, se busca dividir el problema de aprendizaje en diferentes módulos, cada uno enfocado en resolver una parte específica de éste y luego, al acoplarlos, podrían solucionar la tarea completa. Al abordarlo así, se pretende partir la tarea principal de aprendizaje en diferentes sub-tareas. Además, estos modulos podrían trabajar con diferentes versiones de la tarea, ya que no estarán asociados a un videojuego específico, sino a un género entero, por lo tanto, esta arquitectura podría generalizar el procesamiento por sub-tareas para diferentes videojuegos.

1.1. Los videojuegos del género MOBA

Se trata de un género de videojuegos de estrategia en tiempo real en equipo, donde dos equipos, de entre 3 y 5 miembros cada uno, se enfrentan en un mapa para tratar de destruir la base enemiga. La base de cada equipo se compone por un conjunto de estructuras, típicamente torres y la base central. Las torres se distrubyen a lo largo del mapa, el cual se divide en tres líneas principales. En cada línea hay un conjunto de torres de cada equipo, por lo que, para ganar la partida, estas torres deben destruirse para llegar a la base central.

Tabla 1. Relación entre fases y situaciones de juego.

Fase de juego (por nombre)	**Situaciones de juego que pueden suceder (por número)**
Selección y prohibición de personajes	-
Apertura	1, 2, 3, 4, 5
Líneas (laning)	6, 7 ,8
Juego medio	3, 4, 9, 10, 12, 13
Juego tardío	3, 4, 9, 10, 11, 12, 13

En la zona entre líneas o jungla, se reparten diversos sub-objetivos, como enemigos neutrales y objetos, que otorgan beneficios al equipo que los consigue. Los miembros de cada equipo deben elegir un personaje o avatar, que usarán a lo largo de la partida. Los personajes pertencen a uno o varios roles, los cuales determinan el tipo de habilidades que tendrá cada personaje. En los MOBA existen entre 5 y 12 roles, a continuación, se definen los que permanecen constantes a lo largo de todos ellos:

1. Carry o atacante. Su papel es generar mucho daño en poco tiempo a los avatares enemigos.
2. Jungla o ágil. Su objetivo es recorrer la jungla consiguiendo los sub-objetivos presentes en ella y asistir a los demás jugadores en las líneas.
3. Tanque o defensivo. Su rol consiste en defender a las torres y a los jugadores aliados, absorbiendo la mayor cantidad de daño posible en las peleas.
4. Soporte o curador. Su tarea es ayudar al resto del equipo, haciéndolos más fuertes y/o resistentes.
5. Guerrero o todo terreno. Su meta es soportar suficiente daño en las peleas mientras que genera una buena cantidad de daño a los enemigos.

Además del objetivo global de destruir la base enemiga, cada jugador tiene la meta individual de maximizar la experiencia y recursos, los cuales le sirven para aumentar los efectos de sus habilidades. La partida inicia con cada jugador en la base central de su equipo y termina cuando una base central es destruida o cuando se acaba el tiempo límite.

Durante la partida, cada jugador recibe información parcial del mapa, ya que sólo saben lo que hay a un cierto radio de distancia de dónde se encuentra su personaje y lo que hay a cierto radio de las torres, el resto de objetos que permanecen fuera de esos radios, son desconocidos para los jugadores. Además de los avatares, típicamente cada equipo cuenta con un ejército de súbditos o creeps, que son unidades que aparecen cada cierto tiempo en la base principal y que se mueven y atacan al equipo contrario de forma automática.

Cada partida dura entre diez minutos y hora y media, dependiendo del videojuego en cuestión y de la habilidad de cada equipo. Dadas estas condiciones, se ha identificado a este género de videojuegos como una plataforma ideal para probar algoritmos de inteligencia colaborativa, puesto que es necesario hacer predicciones a largo plazo, con información incompleta, mientras que se coordinan las acciones de cada miembro del equipo para lograr la meta global, además, cada jugador busca desempeñarse de acuerdo al rol de su personaje y de cumplir con los objetivos individuales.

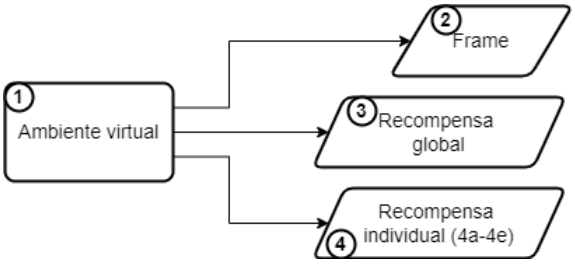

Fig. 1. Diagrama del ambiente virtual.

1.2. Fases y situaciones de juego

En esta sección se describen las fases de una partida estándar de un videojuego MOBA según lo descrito en [2], enlistando cuales son las situaciones de juego particulares que pueden presentarse en cada fase. A continuación, se definen las fases de juego:

1. Selección y prohibición de personajes. En esta fase cada miembro de cada equipo selecciona a su personaje, además, eligen un conjunto de personajes que los rivales no podrán usar en la partida. Es una fase previa a la partida o el juego como tal.
2. Apertura. Esta fase inicia cuando empieza la partida y termina cuando los creeps han llegado a la mitad de cada línea. Esta fase consiste de dos partes, la compra inicial de objetos (en caso de que el videojuego lo incluya) y el posicionamiento de los avatares en las líneas.
3. Líneas (Laning). Esta fase inicia cuando los creeps han llegado a la mitad de cada línea y consiste en quedarse dentro de los límites de la línea tratando de recolectar recursos. Esta fase termina cuando un avatar tiene suficientes recursos para obtener objetos valiosos o tiene una gran diferencia de niveles con respecto al resto de los avatares.
4. Medio juego. Se trata de la fase donde la estrategía toma forma después de la fase de líneas. Esta fase cuenta con desafíos muy diversos, sobresaliendo la cooperación y el razonamiento en tiempo real, donde la sinergia del equipo es vital para definir el rumbo de la partida. Esta fase termina cuando todos los avatareas están en nivel alto y, en caso de que el videojuego lo incluya, sólo les falta comprar uno o dos objetos para equipar.
5. Juego tardío. Es la última fase del juego. Inicia al finalizar el medio juego. En esta etapa, los combates y las estrategias se vuelven más complicados dado que todos los personajes han alcanzado su máximo potencial. Esta fase termina cuando un equipo es derrotado.

Por su parte, las situaciones de juego son los desafíos individuales que pueden presentarse en cada fase de juego. Nótese que las situaciones de juego no tienen un flujo predeterminado, así que aunque se presenten en un cierto orden en la lista, no necesariamente se presentan siempre las mismas en el mismo orden en todas las partidas. Las situaciones de juego son las siguientes:

1. Compra de objetos. Cada jugador interactúa con la tienda y compra objetos de acuerdo a los recursos con los que cuenta y con los beneficios que ese objeto le dará. No hay una definición de victoria o derrota en esta situación.
2. Posicionamiento de los roles. Cada jugador elige, de acuerdo a la estrategia del equipo, una línea (línea superior, línea media, junglas o línea inferior), donde defenderá las estructuras y recolectará recursos. No hay una definición de victoria o derrota en esta situación.
3. Persecución (pickoffs). En esta situación, un avatar es perseguido por uno o varios avatares del equipo contrario, con el fin de matarlo. Esta situación la ganan los perseguidores si logran matar al avatar perseguido y la gana el avatar perseguido si logra escapar de los perseguidores hacia una zona segura (detrás de sus estructuras).
4. Peleas de equipo (team fight). En esta situación dos o más avatares de un equipo pelean contra dos o más avatares del equipo contrario. La victoria la obtiene el equipo cuyos miembros involucrados en la pelea logran matar a todos los miembros del otro equipo involucrados en la pelea y la derrota es para el equipo cuyos miembros involucrados en la pelea son asesinados.
5. Invasión de jungla. Esta situación implica que un personaje, típicamente el rol ágil/jungla, pasa a las zonas del equipo enemigo y obtiene recursos en ella. La situación se gana si el invasor logra robar recursos (dar golpe final a enemigos neutrales o tomar objetos) y salir de la zona invadida. Esta situación se pierde si el invasor no logra robar recursos antes de escapar o si es asesinado mientras invade.
6. Farmeo (farming). En esta situación, los jugadores recopilan recursos a lo largo del mapa, el recurso principal para esto es dar el golpe final a enemigos neutrales o a creeps enemigos. Esta situación se gana cuando se logra dar el golpe final al enemigo seleccionado, de lo contrario, se pierde.
7. Gankeo (ganking). Esta situación se deriva de las persecuciones o de las peleas grupales y consiste en que un avatar que no estaba involucrado en la situación anterior, se una al combate para dar ventaja a su equipo. Esta situación se gana cuando el miembro que se une (ganker) mata al avatar perseguido o logra ganar la pelea en equipo, si este avatar es asesinado, entonces la situación se pierde.
8. Vagar (roaming). Esta situación implica que un personaje, típicamente el ágil/jungla, recorre diversas zonas del mapa para recolectar recursos o en busca de oportunidades de gankeo o para invadir jungla. No hay una definición de victoria o derrota en esta situación.
9. Empujar línea (lane push). En esta situación varios miembros de un equipo atacan una estructura enemiga para destruirla y reducir el espacio seguro del equipo contrario. Esta situación la gana el equipo atacante si logra destruir la estrcutura enemiga y la gana el equipo defensor si logra matar a todos los avatares involucrados en el ataque a la estructura.
10. Empuje dividido (split pushing). En esta situación un miembro de un equipo ataca una estructura enemiga para destruirla y reducir el espacio seguro del equipo contrario. Esta situación la gana el atacante si logra destruir la estrcutura enemiga y la gana el equipo defensor si logra matar al atacante.
11. Cambio de objetos. Esta situación es posterior a la compra de objetos y típicamente sucede sólo cuando un avatar ya no puede comprar más objetos y consiste en que un jugador decide cambiar alguno de los objetos que ya tiene equipados por otros.

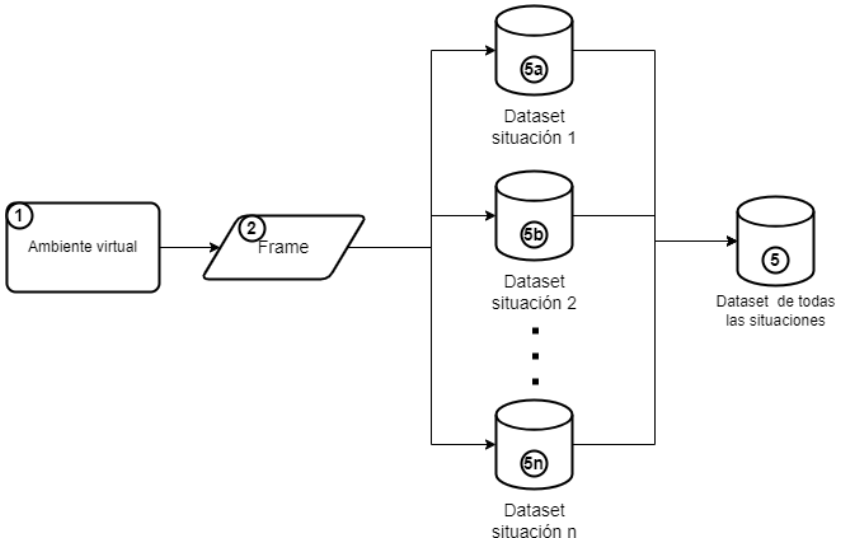

Fig. 2. Diagrama de la generación de datos.

Sólo sucede en videojuegos donde los avatares pueden comprar objetos. No hay una definición de victoria o derrota en esta situación.

12. Sitiar (sieging). Es una situación derivada de empujar líneas y consiste en que el equipo atacante haga retroceder al equipo defensor por detras de la estructura atacada, de modo que son incapaces de defenderla y el equipo atacante puede destruir la estructura más fácilmente. Esta situación la gana el equipo atacante si logra destruir la estrcutura enemiga y la gana el equipo defensor si logra matar a todos los avatares involucrados en el ataque a la estrcutura.
13. Emboscada. En esta situación uno o más miembros de un equipo se ocultan en zonas que bloquean la visión enemiga (arbustos, neblina, etc.) y esperan a que uno o varios avatares enemigos pasen cerca del escondite para atacarlos y matarlos. Esta situación la gana el equipo emboscante si logra matar a los avatares emboscados y la pierden si el equipo emboscado mata a los emboscantes o logra escapar a una zona segura (detrás de sus estructuras).

Finalmente, las situaciones de juego pueden suceder en una o varias fases de juego, esta relación se muestra en el Tabla 1.

2. Estado del arte

Los agentes jugadores son un tema de especial interés para la IA, en el caso de los videojuegos del género MOBA, éste se intensifica debido a la complejidad que presentan los desafíos de estos videojuegos, considerándose como un género que requiere de ciertos rasgos de inteligencia, como trabajo en equipo, coordinación, predicción a corto y largo plazo, inferencia con información incompleta, entre otros.

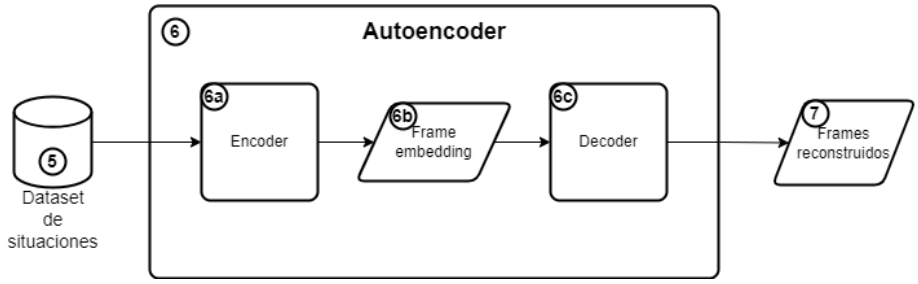

Fig. 3. Diagrama del entrenamiento del Autoencoder profundo.

Para fines de esta investigación, los trabajos más relevantes sobre agentes jugadores son precisamente aquellos sobre equipos jugadores de MOBA. De estos trabajos, sobresalen tres investigaciones, la primera es el trabajo [3], donde presentan una arquitectura basada en lógica difusa para agentes jugadores de MOBA, haciendo una simplificación del comportamiento del bot con una máquina de estados difusa con tres estados.

Este enfoque busca una respuesta sencilla, en cuanto a complejidad computacional, para resolver los MOBA, sin embargo, se probó en un bot único, no en un equipo. La segunda investigación, se encuentra en los trabajos [4, 5], cuyo trabajo derivó en el primer equipo de bots jugadores de DoTA2 capaz de derrotar equipos profesionales.

La relevancia de este trabajo recae en los hallazgos que tuvieron de la red una vez entrenada, puesto que, aunque no se programó específicamente para detectar situaciones, los estudios de interpretabilidad que hicieron a su arquitectura reforzada profunda, revelaron que el algoritmo era capaz de detectar algunas situaciones y reaccionar adecuadamente en consecuencia a ellas.

La tercera, es la investigación [6], cuyos autores desarrollaron un algoritmo de aprendizaje supervisado, donde se asocian estados del mapa con comportamientos que, dado el análisis que se hizo del conjunto de datos, estaba más asociado con victorias. Con este algoritmo lograron obtener un equipo de agentes jugadores de Honor of Kings con desempeño superior al de un equipo humano promedio.

Además de los agentes jugadores, es importante mencionar los avances en investigación sobre agentes jugadores generales, es decir, aquellos que juegan varios juegos una vez entrenados. En esta área, destaca el trabajo [7], en el cual buscan entrenar un agente reforzado que generalice su capacidad de jugar juegos (en inglés se le conoce como General Game Playing o GGP). Para este trabajo se experimenta la transferencia de conocimiento para que agentes inteligentes logren resolver videojuegos 1vs1, donde se evalúa al agente con la arquitectura que proponen contra el algoritmo del estado del arte Upper Confidence Bound on Trees (UCT).

En otros trabajos relacionados con el esquema propuesto en este artículo se usaron autoencoders como extractores de características profundas para agentes jugadores y generadores de contenido, como los presentados en [8, 9], respectivamente. Por otro lado, la idea de utilizar ambientes virtuales de entrenamiento en busca de la generalización se puede rescatar de las investigaciones presentadas en [10, 11, 12].

La diferencia principal de este trabajo con el estado del arte, se encuentra en el planteamiento modular del problema de aprendizaje, separando el comportamiento de equipo para jugar videojuegos MOBA en: análisis de la imágen del mapa para extracción de características; clasificación de estados de juego para el ajuste del algoritmo reforzado; el algoritmo reforzado no profundo para jugar MOBAs, que puede ser no profundo gracias a los dos módulos anteriores; el entrenamiento en ambientes virtuales, diferentes a los ambientes reales.

Además del planteamiento modular del problema, por medio del extractor de características, se lograría crear una arquitectura para un equipo que juegue diversos videojuegos MOBAs, puesto que aprendería a resolver las diferentes situaciones de juego independientemente del videojuego del género MOBA.

3. Desarrollo de la solución

3.1. Planteamiento del aprendizaje de un comportamiento colaborativo por situaciones de juego

Para resolver el problema global de aprender un comportamiento adecuado para jugar videojuegos MOBA se propone resolver cada una de sus partes, identificadas como situaciones de juego. Para ello, se plantea el uso de un ambiente virtual, en donde se presenten condiciones propias de cada situación de juego a un equipo de agentes inteligentes.

El ambiente virtual debe presentar al equipo condiciones que propicien el aprendizaje, por lo tanto, no será precisamente como algún ambiente real (un videojuego MOBA real), sino un pseudo-videojuego donde los desafíos sean lo suficientemente similares a los de los ambientes reales como para poder extrapolar el conocimiento adquirido a problemas desconocidos parecidos, pero lo suficientemente diferente para poner a prueba la capacidad de generalización del modelo entrenado. Por lo tanto, la primera parte del esquema de aprendizaje es el ambiente virtual, presentado en la Figura 1.

En el bloque 1 se ejecuta el motor del pseudo-videojuego, el cual debe calibrarse para presentar una situación de juego en particular a partir de una definición formal de las situaciones de juego, siendo este el espacio idóneo para aprender esa situación. Como resultado, este módulo produce tres datos, el primero de ellos, el bloque 2, es el Frame, que es la matriz asociada a lo que se mostraría en pantalla, este dato es la fuente principal de información sobre el pseudo-videojuego, ya que es el equivalente a la información disponible para los jugadores en los videojuegos MOBA.

Además de la matriz, este ambiente debe informar al equipo sobre su desempeño para ganar la situación de juego a la que se enfrente, esta información se proporciona mediante las recompensas en los bloques 3 y 4. El bloque 3 le indica al equipo completo qué tan bueno es su comportamiento para resolver la situación de juego y el bloque 4 se divide en cinco datos, una recompensa para cada rol y miembro de un equipo estándar de un videojuego MOBA, estas recompensas, de la 4a a la 4e, le indican a cada agente qué tan bien cumplen con su rol en la situación de juego a la que se enfrentan. Ambas recompensas, toman valores en el rango [0,1], donde 0 es el peor desempeño posible y 1 es el mejor.

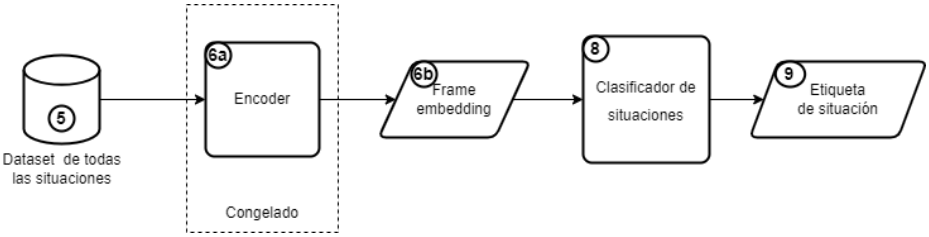

Fig. 4. Diagrama del entrenamiento del clasificador de sitaciones.

El ambiente será el módulo a través del cual se obtendrán los frames asociados a situaciones de juego específicas, por lo tanto, es posible aprovecharlo como mecanismo generador de datos etiquetados. Por ello, como se muestra en la Figura 2, los frames generados en el ambiente virtual se almacenarán, en conjuntos por cada situación, en los bloques 5a hasta el 5n, correspondiendo a las n situaciones que se hayan definido. Esta información se combinaría posteriormente en un conjunto que contenga todas las situaciones, en el bloque 5.

Para emplear sólo la información visual más importante del frame, se propone un mecanismo enfocado únicamente a la extracción de características. Esto es posible mediante el uso de un autoencoder profundo (Deep Autoencoder, DAE), cuyo entrenamiento no supervisado se presenta en la Figura 3. El DAE (bloque 6), puede entrenarse una vez que el bloque 5 tenga información de al menos una de las n situaciones de juego definidas.

El bloque 5 enviará frames de las situaciones de juego a la aqruitectura del DAE en el bloque 6. Este bloque se divide en tres partes, el primero es el encoder (6a), que es la parte que extrae características de los frames y produce un vector de dimensiones reducidas (frame embedding), este vector es el dato del bloque 6b. Este frame embedding se usa en el decoder (6c), para reconstruir la imagen de entrada, que corresponde al dato del bloque 7. Los frames reconstruidos se usan úncamente para evaluar el aprendizaje del DAE, entre más similares sean a los frames de entrada, mejor desempeño tiene el DAE.

Ya que el DAE ha sido entrenado, entonces es posible utilizar los frame embeddings que produce el bloque 6a en fases posteriores. Además de un mecanismo para determinar la información visual más relevante, el equipo requiere de un módulo que le ayude a determinar a qué situación de juego en particular se está enfrentando. Para ello, se propone un clasificador de situaciones, mismo que podría entrenarse de manera supervisada, dado que en el conjunto de datos del bloque 5 cada imágen estaría asociada con una situación de juego particular.

El entrenamiento de este clasificador se muestra en la Figura 4, donde el dataset completo (bloque 5), enviaría frames al bloque 6a, el encoder pre-entrenado y congelado, que generaría un frame embedding por cada imagen del dataset (bloque 6b). Estos embeddings serían la entrada al clasificador (bloque 8). Este bloque produce como salida una etiqueta de situación de juego (bloque 9), la cual, una vez que el clasificador haya sido entrenado, debería ser igual a la etiqueta original de las imágenes del dataset.

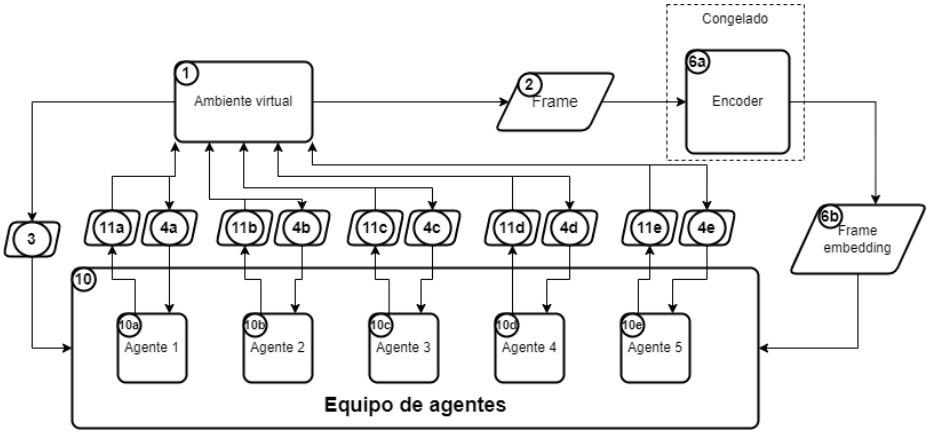

Fig. 5. Diagrama del entrenamiento de una política del equipo de agentes inteligentes para resolver una situación de juego.

En cuanto al entrenamiento del equipo de agentes para resolver una situación de juego en particular, se propone el esquema de la Figura 5. Este esquema inicia con el bloque 1, el ambiente virtual, el cual envía el frame actual del pseudo-videojuego (bloque 2), al codificador pre-entrenado (bloque 6a), éste produce un frame embedding (bloque 6b), que será la entrada para los agentes en el bloque 10, mismo que se divide en los bloques del 10a al 10e, uno por cada miembro y rol de un equipo estándar de un videojuego MOBA.

Todos los agentes reciben el bloque 6b y lo usan para elegir una acción de acuerdo a su política. Dada una situación i de juego, los agentes emplearán una política i correspondiente. En esta etapa, el ambiente generaría sólo espacios que presenten una situación de juego específica, por lo tanto, los agentes pueden ajustar únicamente la política que corresponde a dicha situación.

Cada agente produce como respuesta al frame embedding, una acción, con la que buscan obtener la victoria o terminar la situación de juego actual. Las acciones de los agentes se envían al ambiente por medio de los bloques del 11a al 11e, donde cada uno de ellos está asociado a un agente en particular.

En conjunto, todas las acciones producen un único cambio en el ambiente, que produce un nuevo frame en consecuencia. Para indicarle al equipo qué tan bueno fue su desempeño, el ambiente envía las recompensas, la global en el bloque 3 y la individual en los bloques del 4a al 4e. Los agentes usarán esas recompensas para ajustar su política, la cual debe ayudarlos a resolver la situación de juego que están aprendiendo en ese momento dadas las condiciones actuales del ambiente, presentes en el frame embedding.

En este esquema, los agentes están aprendiendo a resolver cada parte de la tarea global, es decir, ajustan una política que sirve para resolver una situación de juego, aprendiendo cada situación por separado en vez de aprender a jugar un videojuego MOBA, sin embargo, para poder resolver una partida completa se presenta el esquema de la Figura 6.

En éste, la ejecución comienza en el bloque 14, que sustituye al bloque 1, este bloque correponde a un videojuego MOBA, el cual únicamente produce el frame (bloque 2) (nótese que este bloque 2 es equivalente, pero no igual al bloque 2 de las figuras anteriores, las dimensiones y los valores en él pueden variar), que se envía hacia el codificador pre-entrenado (bloque 6a), que produce el frame embedding (6b), el cual llega al clasificador de situaciones pre-entrenado (bloque 8) y al equipo de agentes (bloque 10).

En este esquema, las situaciones irán cambiando, ya que se trata de una partida completa de un MOBA, por ello, para poder determinar cuál de todas las políticas pre-ajustadas deben usar los agentes, se utiliza el clasificador de situaciones, el cual emitirá una etiqueta de situación de juego (bloque 9), que será utilizada por un selector de políticas (bloque 12), el cual será un mecanismo que le indique a los agentes qué política usar, de acuerdo a la situación de juego identificada en la etiqueta del bloque 9.

El selector de políticas del bloque 12 enviará una señal (bloque 13), a cada agente, del 10a al 10e, para que empleen la política correspondiente a la situación de juego actual. Cada agente generará una acción de acuerdo a la política seleccionada y a la información actual de la partida contenida en el frame embedding.

Estas acciones, enviadas al videojuego MOBA a traves de los bloques del 11a al 11e, generarán un cambio en la partida y por lo tanto, el frame producido por el bloque 14 contiene este cambio producido a consecuencia de las acciones de los agentes.

En la Figura 6, todos los bloques basados en aprendizaje máquina han sido pre-entrenados, el codificador de manera no supervisada, el claisficador bajo el enfoque supervisado y el equipo de agentes bajo un enfoque reforzado, por lo que en este esquema no hay necesidad de hacer el ajuste de ningún modelo.

Sin embargo, para que el equipo de agentes logre resolver diferentes versiones de videojuegos MOBA, es necesario tener una definición general de todos los videojuegos que sea comprensible por los agentes. Para lograrlo, se propone entrenar diversos codificadores, uno por cada videojuego MOBA que deba resolver el equipo de agentes. Estos codificadores se entrenarían tomando como base un dataset de imágenes de un videojuego MOBA en particular, mismo que se obtendría de manera similar al dataset de situaciones de juego, presentado en la Figura 2.

Con estos datasets específicos para cada videojuego, es posible entrenar diversos codificadores, uno por cada juego, un encoder generaría frame embeddings de las mismas dimensiones que los del bloque 6b, puesto que serían la entrada del decoder 6c, el que se entrenó para reconstruir imágenes del ambiente virtual de entrenamiento, que quedan plasmadas en el bloque 7.

De este modo, todos los encoders se entrenarían para producir embeddings que sirvan para reconstruir imágenes del ambiente virtual, como si todos los ambientes se tradujeran a características del ambiente donde fue entrenado el equipo de agentes inteligentes. Este entrenamiento de encoders para cada MOBA y su correcto uso durante la fase de ejecución en el diagrama de la Figura 6 son el mecanismo de generalización que se propone para resolver diversos videojuegos del género MOBA.

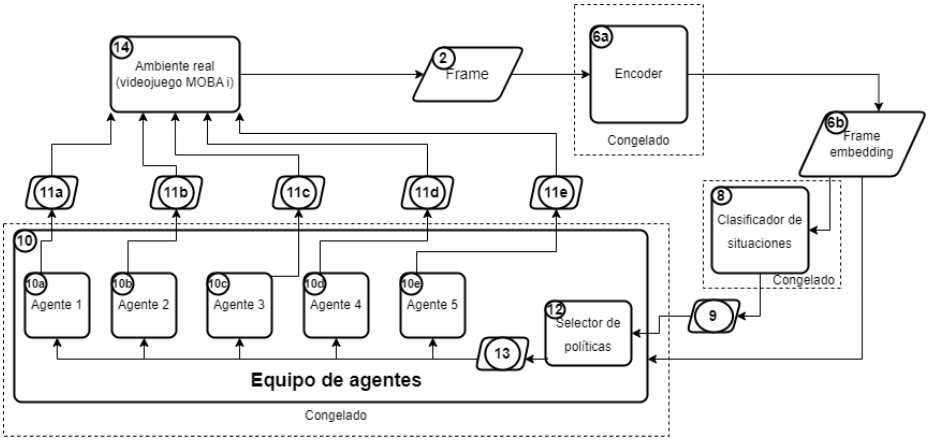

Fig. 6. Diagrama para una partida de un videojuego MOBA por situaciones de juego.

3.2. Situaciones complejas vs. situaciones triviales

El esquema propuesto en la sección anterior se propone para aquellas situaciones de juego que implican un proceso complejo de toma de decisiones individuales y de estrategia en equipo, sin embargo, existen situaciones de juego que no son tan complejas y no requieren de un procesamiento tan profundo de la información para tomar las decisiones.

Hay situaciones en las que, por ejemplo, los jugadores solo deben moverse hacia algún punto en particular, como al inicio de cada partida cuando cada avatar se posiciona en una línea cerca de alguna torre o estructura. Para este tipo de situaciones, se pueden usar políticas sencillas como instrucciones condicionales (if) en los agentes, sin necesidad de usar aprendizaje reforzado.

3.3. Novedad científica

El ensamble propuesto en este trabajo no se ha encontrado en el estado del arte, sin embargo, más que el esquema nuevo, la aportación viene desde el plantemiento modular con base en el aprendizaje de situaciones de juego. El esquema es la forma de materializar la propuesta de solución que se da con este análisis del problema. Además, el esquema propuesto busca generalizar su aprendizaje para resolver varios videojuegos del mismo género, problema que tampoco se ha resuelto en el estado del arte.

3.4. Evaluación

Aunque no existe una propuesta exactamente igual a la de este artículo, sí es posible hacer comparaciones del desempeño del equipo de agentes jugadores de MOBA. En primer lugar, existen equipos de agentes artificiales capaces de resolver juegos MOBA (principalmente DoTA 2, League of Legends y Honor of Kings), por lo que a nivel de una versión de la tarea, es posible comparar el desempeño del equipo propuesto.

Esto se mide a partir de variables bien definidas, como el índice de desempeño, el KDA o el porcentaje de victorias. Estas medidas de desempeño para cada caso servirían para comparar el desempeño del equipo de agentes jugadores propuesto con otros equipos que resuelven un único videojuego. Por otro lado, para medir qué tan bueno es el equipo para generalizar, se planea usar esas mismas métricas, aplicadas para los diferentes videojuegos.

Es decir, se tomarían las medidas de esas variables para el mismo equipo resolviendo diferentes videojuegos MOBA, una vez que ya ha sido entrenado en el ambiente virtual. De esto modo se corroboraría la capacidad del equipo para resolver situaciones similares a las que aprendió durante el entrenamiento pero que no son exactamente iguales, dado que algunas condiciones y/o reglas varían en los diferentes videojuegos.

4. Conclusiones y trabajo futuro

En este artículo se presenta un esquema de aprendizaje híbrido que busca definir una estrategia de aprendizaje para agentes jugadores capaces de resolver distintos videojuegos del género MOBA. Es necesario llevar a cabo los experimentos, para contrastar el planteamiento teórico de la propuesta de solución con los resultados en implementación para comprobar que realmente el esquema híbrido propuesto es capaz de resolver la tarea como se ha planteado que lo hará. El planteamiento presentado no contempla la posibilidad de que varias situaciones de juego puedan presentarse de manera simultánea, por lo que incluir esta condición al problema puede formar parte de una nueva investigación.

Agradecimientos. Este trabajo fue posible gracias al apoyo del gobierno mexicano a través del programa FORDECYT-PRONACES del Consejo Nacional de Ciencia y Tecnología (CONACYT) bajo la beca APN2017–5241; las becas de investigación de la SIP-IPN SIP-2259, SIP-20231198 Y SIP-20230140; la IPN-COFAA y la IPN-EDI.

Referencias

1. Lindqvist, K., Nilsson, D.: Developing a 5v5 framework for DOTA 2 bot competition (2020)
2. Silva, V., do N., Chaimowicz, L.: MOBA: A new arena for game artificial intelligence (2017) doi: 10.48550/ARXIV.1705.10443
3. Waltham, M., Moodley, D.: An analysis of artificial intelligence techniques in multiplayer online battle arena game environments. In: Annual Conference of the South African Institute of Computer Scientists and Information Technologists, no. 17, pp. 1–7 (2016) doi: 10.1145/2987491.2987513
4. Raiman, J., Zhang, S., Wolski, F.: Long-term planning and situational awareness in OpenAI five (2019) doi: 10.48550/ARXIV.1912.06721
5. Berner, C., Brockman, G., Chan, B., Cheung, V., Debiak, P., Dennison, C., Farhi, D., Fischer, Q., Hashme, S., Hesse, C., Józefowicz, R., Gray, S., Olsson, C., Pachocki, J., Petrov, M., Pinto, H.P. d. O., Raiman, J., Salimans, T., Schlatter, J., Schneider, et al.: DOTA 2 with large scale deep reinforcement learning (2019) doi: 10.48550/ARXIV.1912.06680

6. Ye, D., Chen, G., Zhao, P., Qiu, F., Yuan, B., Zhang, W., Chen, S., Sun, M., Li, X., Li, S., Liang, J., Lian, Z., Shi, B., Wang, L., Shi, T., Fu, Q., Yang, W., Huang, L.: Supervised learning achieves human-level performance in MOBA games: A case study of honor of kings. In: IEEE Transactions on Neural Networks and Learning Systems, vol. 33, no. 3, pp. 908–918 (2020) doi: 10.48550/ARXIV.2011.12582
7. McEwan, C., Thielscher, M.: Knowledge transfer for deep reinforcement agents in general game playing. In: Australasian Joint Conference on Artificial Intelligence, vol. 13151, pp. 53–66 (2022) doi: 10.1007/978-3-030-97546-3
8. Alvernaz, S., Togelius, J.: Autoencoder-augmented neuroevolution for visual doom playing. In: IEEE Conference on Computational Intelligence and Games, pp. 1–8 (2017) doi: 10.48550/ARXIV.1707.03902
9. Jadhav, M., Guzdial, M.: Tile embedding: A general representation for level generation. In: Conference on Artificial Intelligence and Interactive Digital Entertainment, vol. 17, no. 1, pp. 34–41 (2021) doi: 10.1609/aiide.v17i1.18888
10. Dubey, R., Agrawal, P., Pathak, D., Griffiths, T. L., Efros, A. A.: Investigating human priors for playing video games (2018) doi: 10.48550/ARXIV.1802.10217
11. Open Ended Learning Team, Stooke, A., Mahajan, A., Barros, C., Deck, C., Bauer, J., Sygnowski, J., Trebacz, M., Jaderberg, M., Mathieu, M., McAleese, N., Bradley-Schmieg, N., Wong, N., Porcel, N., Raileanu, R., Hughes-Fitt, S., Dalibard, V., Czarnecki, W. M.: Open-ended learning leads to generally capable agents (2021) doi: 10.48550/ARXIV.2107.12808
12. Reed, S., Zolna, K., Parisotto, E., Colmenarejo, S. G., Novikov, A., Barth-Maron, G., de Freitas, N.: A generalist agent (2022) doi: 10.48550/ARXIV.2205.06175