# Cooperación emergente mediante aprendizaje profundo por refuerzo

por

Daniel Santiago Cuervo Gómez

Presentado a la Facultad de Ingeniería
en cumplimiento parcial de los requisitos para optar al título de

Ingeniero Electrónico

de la

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS

Autor .....
Facultad de Ingeniería
Octubre del 2019

Supervisado por .....
Marco Aurelio Alzate Monroy
Profesor Titular
Director

Aceptado por .....
Miguel Alberto Melgarejo Rey
Profesor Asociado
Jurado

# Cooperación emergente mediante aprendizaje profundo por refuerzo

por

Daniel Santiago Cuervo Gómez

## Resumen

En este trabajo se presenta el diseño de un sistema multi-agente compuesto de redes neuronales y sintonizado mediante algoritmos de *aprendizaje profundo por refuerzo*. El sistema es aplicado a un *dilema social*, un problema cuya solución óptima requiere que los agentes coordinen sus acciones para maximizar una función de desempeño macroscópica, a pesar de que los objetivos individuales de cada agente son divergentes con este objetivo global.

Los agentes aquí considerados son *egoístas*, es decir, su objetivo es maximizar su desempeño individual sin considerar el desempeño global del sistema, por lo que la cooperación, el objetivo del diseño, será un fenómeno emergente. Para lograr tal fin, se propone el uso de una función objetivo que busca promover la coordinación entre los agentes mediante la maximización de la *información mutua* entre sus acciones.

El sistema propuesto se compone de agentes homogéneos, y cada agente se compone de varias redes neuronales que implementan sus funciones de percepción y acción. La información mutua es estimada también usando una red neuronal, de modo que se tiene una representación diferenciable de esta que se suma a la función objetivo de un algoritmo de gradiente ascendente para sintonizar los componentes del agente de modo que se maximice su desempeño individual, y la información mutua entre sus acciones y las acciones del resto de la población.

Para evaluar el diseño se definen un conjunto de índices macroscópicos de desempeño y cooperación y se comparan según estos los resultados obtenidos al aplicar al problema el sistema propuesto, y un sistema sin el objetivo de maximización de la información mutua, obteniéndose resultados significativamente superiores en el primero. Ante esto se concluye que la maximización de la información mutua entre agentes promueve la emergencia de fenómenos cooperativos en un problema con las características de un dilema social.

La metodología de diseño del sistema, al ser completamente descentralizada, podría solventar diversas dificultades de muchos de los algoritmos actualmente existentes para el diseño de sistemas multi-agente neuronales cooperativos, como las altas demandas de ancho de banda y memoria que dificultan su aplicación práctica.

**Director**: Marco Aurelio Alzate Monroy, Profesor Titular.

# Índice general

# Notación

## Notación General

### Variables

Las letras minúsculas en negrilla ($\mathbf{a}$) son variables aleatorias, las letras minúsculas ($a$) representan valores que toma una variable aleatoria o cantidades escalares. Una flecha sobre el nombre de la variable ($\vec{a}$ o $\vec{\mathbf{a}}$) indica que la variable es un vector columna. Las letras mayúsculas en negrilla ($\mathbf{A}$) son matrices[1]. Las letras caligráficas ($\mathcal{A}$) son conjuntos. Las letras mayúsculas ($A$), así como las letras griegas $\pi$, $\phi$, y $\zeta$, son usadas para representar funciones.

### Conjuntos

| | |
|---|---|
| $\mathbb{R}$ : | conjunto de los números reales. |
| $\mathbb{N}$ : | conjunto de los números naturales. |
| $\mathbb{Z}$ : | conjunto de los números enteros. |
| $\mathbb{R}^n$ : | conjunto de los vectores reales de dimensión $n$. |
| $\mathbb{R}^{m\times n}$ : | conjunto de las matrices reales de $m$ filas y $n$ columnas. |
| $\{0,1\}$ : | conjunto que contiene los elementos 0 y 1. |
| $\{0,1,...,n\}$ : | conjunto de todos los enteros entre 0 y $n$. |
| $[a,b)$ : | intervalo real entre $a$ y $b$ incluyendo $a$. |
| $[a,b]^n$ : | vector de dimensión $n$ cuyos elementos están en el intervalo $[a,b]$. |

[1] Que pueden ser variables aleatorias.

| | |
|---|---|
| $[a, b]^{m \times n}$ : | matriz de $m$ filas y $n$ columnas cuyos elementos están en el intervalo $[a, b]$. |
| $\forall a \in \mathcal{A}$ : | se lee como *"para todo elemento a que pertenece al conjunto* $\mathcal{A}$". |
| $\lvert\mathcal{A}\rvert$ : | cardinalidad del conjunto $\mathcal{A}$. |
| $\mathcal{A} \times \mathcal{B}$ : | producto cartesiano entre el conjunto $\mathcal{A}$ y el conjunto $\mathcal{B}$. |

### Indexación

| | |
|---|---|
| $\vec{a}_i$: | $i$-ésimo elemento del vector $\vec{a}$, indexando desde $i = 1$. |
| $\mathbf{A}_{i,j}$: | elemento de la $i$-ésima fila y la $j$-ésima columna de la matriz $\mathbf{A}$. |
| $\mathbf{A}_{i,:}$: | $i$-ésima fila de la matriz $\mathbf{A}$. |
| $\mathbf{A}_{:,i}$: | $i$-ésima columna de la matriz $\mathbf{A}$. |

### Funciones

| | |
|---|---|
| $F : \mathcal{A} \to \mathcal{B}$: | función $F$ con dominio $\mathcal{A}$ y rango $\mathcal{B}$. |
| $F(x; \theta)$: | función $F$ parametrizada por $\theta$. |
| $\mathbb{1}(x; \text{condición})$: | función indicador de $x$, cuyo valor es 1 si la condición se cumple, o 0 en caso contrario. |
| máx$(a; b)$: | función que toma el valor más grande de los posibles valores $a$ y $b$. |
| $\text{argmax}_a F(a)$: | valor de $a$ para el que la función $F(a)$ toma el valor máximo. |
| $\text{máx}_a F(a)$: | valor máximo de $F(a)$ a medida que $a$ varía en un dominio. |
| $\text{mín}_a F(a)$: | valor mínimo de $F(a)$ a medida que $a$ varía en un dominio. |

Cuando una función $F$ cuyo dominio son escalares se aplica a un vector $F(\vec{a})$ o matriz $F(\vec{\mathbf{A}})$, se interpreta que la función se aplica a cada elemento del arreglo. Por ejemplo, si $\vec{b} = F(\vec{a})$, entonces $\vec{b}_i = F(\vec{a}_i)$, para todo $i$.

### Cálculo

| | |
|---|---|
| $\nabla_\theta f(a)$: | gradiente de la función $f$ evaluada en $a$ respecto a la variable $\theta$. |
| $\sum_{i=1}^{n} x_i$: | suma de todos los valores de $x_i$ desde $i = 1$ hasta $i = n$. |

$\sum_i x_i$: suma de todos los posibles valores de $x_i$.

### Álgebra lineal

$\vec{b}^{(i)}$: denota la $i$-ésima base canónica (un vector nulo salvo el $i$-ésimo elemento cuyo valor es 1).

$\vec{a}^T$: vector $\vec{a}$ transpuesto.

$\lambda$: valores propios de una matriz.

### Probabilidad

| | |
|---|---|
| $P_{\mathbf{a}}(a) = p(\mathbf{a} = a)$: | probabilidad de que la variable aleatoria $\mathbf{a}$ tome el valor $a$. |
| $P_{\mathbf{ab}}(a, b) = p(\mathbf{a} = a, \mathbf{b} = b)$: | probabilidad conjunta de que ocurran el evento $\mathbf{a} = a$ y el evento $\mathbf{b} = b$. |
| $P_{\mathbf{a} \mid \mathbf{b}}(a \mid b) = p(\mathbf{a} = a \mid \mathbf{b} = b)$: | probabilidad condicional del evento $\mathbf{a} = a$ dado que ocurrió el evento $\mathbf{b} = b$. |
| $\mathbf{a} \sim P$: | indica que la variable aleatoria $\mathbf{a}$ se distribuye de acuerdo a la distribución de probabilidad $P$. |
| $\mathbb{E}_{\mathbf{a} \sim P}[f(a)]$: | valor esperado de $f(a)$ respecto a $P$. |

# Notación por capítulos

### Sección 2.2

$t$: índice de tiempo discreto.

$L$: horizonte (longitud de un episodio).

$\mathbf{s}_t$: estado del entorno en el instante $t$.

$\mathcal{S}$: conjunto de todos los posibles estados.

$\mathbf{a}_t$: acción ejecutada por el agente en el instante $t$.

| | |
|---|---|
| $\mathcal{A}$: | conjunto de todas las posibles acciones. |
| $T$: | función de transición de estado. |
| $\mathbf{r}_t$: | recompensa obtenida en el instante $t$. |
| $R$: | función de recompensa. |
| $\mathcal{R}$: | conjunto de todas las posibles recompensas. |
| $\pi$: | política del agente. |
| $G(\mathbf{s};\pi)$: | ganancia a largo plazo del agente a partir del estado $\mathbf{s}$ dado que se sigue la política $\pi$. |
| $\pi^*$: | política óptima. |
| $V(s;\pi)$: | función de valor. Valor esperado de ganancia a largo plazo del agente a partir del estado $s$ dado que se sigue la política $\pi$. |
| $s$: | un valor de la variable aleatoria $\mathbf{s}_t$. |
| $s'$: | un valor de la variable aleatoria $\mathbf{s}_{t+1}$. |
| $Q(s,a;\pi)$: | función de calidad. Valor esperado de ganancia a largo plazo del agente dado que en el estado $s$ se toma la acción $a$, y en adelante se sigue la política $\pi$. |
| $a$: | un valor de la variable aleatoria $\mathbf{a}_t$. |
| $J_\pi$: | función objetivo a maximizar en un algoritmo basado en el gradiente de la política. |
| $\alpha$: | tasa de aprendizaje en un algoritmo basado en el gradiente de la política. |
| $\mathcal{E}$: | trayectoria de experiencias $(s_t, a_t, r_{t+1})$ de longitud $L$. |
| $\pi(a,s;\omega_\pi)$: | aproximador funcional de la política estocástica $p(\mathbf{a}_t = a \vert \mathbf{s}_t = s)$ parametrizado por $\omega_\pi$. |
| $A(s,a)$: | función de ventaja. Diferencia entre $Q(s,a;\pi)$ y $V(s;\pi)$. |
| $\hat{V}(s;\omega_V)$: | aproximador funcional de la función de valor parametrizado por $\omega_V$. |
| $\hat{A}$: | estimado de la función de ventaja usando $\hat{V}$. |
| $J_V$: | función objetivo a minimizar para aproximar la función de valor. |

**Sección 2.3**

| | |
|---|---|
| $\mathcal{A}$: | conjunto de todas las posibles acciones conjuntas. |
| $\mathcal{A}_i$: | conjunto de todas las posibles acciones del $i$-ésimo agente. |
| $N$: | número de agentes en un sistema multi-agente. |
| $\vec{\mathbf{a}}$: | acción conjunta de la población. |
| $m_i$: | número de estrategias del $i$-ésimo agente. |
| $\mathbf{U}$ : | matriz de utilidad en un juego matricial. |
| $c$: | acción cooperativa. |
| $\neg c$: | acción no cooperativa. |
| $G_R$: | resultado de cooperación mutua en un dilema social. |
| $G_P$: | resultado de egoísmo mutuo en un dilema social. |
| $G_S$: | resultado de cooperar con un agente egoísta en un dilema social. |
| $G_T$: | resultado de explotar a un agente cooperativo en un dilema social. |
| $V_i^{\pi^1,\pi^2}(s)$: | valor del estado $s$ para el $i$-ésimo agente, $i \in \{1,2\}$, si el agente 1 sigue una política $\pi_1$ y el agente 2 una política $\pi_2$. |
| $\pi^c$: | una política cooperativa. |
| $\pi^{\neg c}$: | una política no cooperativa. |
| $\vec{\pi}$: | política conjunta. $\vec{\pi} = [\pi_1, ..., \pi_N]$. |
| $\mathcal{M}$: | un juego de Markov. |
| $\Pi^c$: | conjunto de todas las políticas cooperativas. |
| $\Pi^{\neg c}$: | conjunto de todas las políticas no cooperativas. |
| $n_c$: | número de posibles valores (clases) de un píxel en el juego de la cosecha. |
| $c_i$: | valor (clase) de un píxel en el juego de la cosecha. |
| $\vec{\mathbf{a}}$: | acción conjunta de la población. |
| $n_{mzn}$: | número de manzanas en una determinada región en el juego de la cosecha. |

$d$: número de píxeles en una observación del juego de la cosecha.

$\mathcal{O}$: conjunto de todas las posibles observaciones.

$O$: función de observación.

$\mathcal{X}$: conjunto de todas las posibles posiciones en el plano cartesiano discreto.

$\mathbf{S}_t$: estado del entorno en el instante $t$ en el juego de la cosecha (una matriz).

$(\mathbf{x}, \mathbf{y})$: posición del agente en el plano cartesiano discreto.

**Sección 2.5**

$I(\mathbf{x}, \mathbf{y})$: información mutua entre las variables aleatorias $\mathbf{x}$ y $\mathbf{y}$.

$H(\mathbf{x})$: entropía de la variable aleatoria $\mathbf{x}$.

$H(\mathbf{x}|\mathbf{y})$: entropía condicional de la variable aleatoria $\mathbf{x}$ dada la variable aleatoria $\mathbf{x}$.

$D_{KL}(P_{\mathbf{z}}||Q_{\mathbf{z}})$: divergencia de Kullback-Leibler entre las distribuciones de probabilidad $P_{\mathbf{z}}$ y $Q_{\mathbf{z}}$.

$F$: discriminador entre muestras de la distribución $P_{\mathbf{xy}}$ y muestras de la distribución $P_{\mathbf{x}}P_{\mathbf{y}}$ en el estimador de Donsker-Varadhan de la información mutua.

$\Omega$: espacio muestral de las variables aleatorias en el estimador de Donsker-Varadhan de la información mutua.

$F(x, y; \mathbf{W}_F)$: discriminador $F$ implementado como una red neuronal parametrizada por $\mathbf{W}_F$.

$\zeta(x)$: función softplus, $\zeta(x) = \ln(1 + e^x)$.

# Capítulo 3

## Sección 3.1

| | |
|---|---|
| $C(\mathbf{O}; \mathbf{W}_C)$: | sensor. Red neuronal parametrizada por $\mathbf{W}_C$ que recibe una observación y genera una representación comprimida de ésta. |
| $\vec{\mathbf{z}}$: | código generado por el sensor. |
| $D(\vec{\mathbf{z}}; \mathbf{W}_D)$: | decodificador del sensor. Red neuronal parametrizada por $\mathbf{W}_D$ que recibe el código $\vec{\mathbf{z}}$ y define para cada píxel una distribución de probabilidad de pertenencia a una de las $n_c$ clases del juego de la cosecha. |
| $\hat{\mathbf{O}}$: | observación reconstruida a partir de la salida del decodificador del sensor. |
| $J_{C,D}$: | función objetivo a minimizar en el entrenamiento del sensor. |
| $\mathcal{D}$: | conjunto de observaciones muestreadas uniformemente de $\mathcal{O}$ usadas para entrenar el sensor. |
| $\mathbf{C}$: | matriz cuyas $n_c$ columnas de la $i$-ésima fila corresponden a las probabilidades de pertenencia a cada clase para el $i$-ésimo píxel. |
| $\alpha_C$: | tasa de aprendizaje del sensor. |
| $\vec{\mathbf{h}}$: | estado de la memoria del agente. |
| $M(\vec{\mathbf{z}}_t, \vec{\mathbf{h}}_{t-1}; \mathbf{W}_{M_I}, \mathbf{W}_{M_H})$: | memoria del agente. Red neuronal recurrente parametrizada por $\mathbf{W}_{M_I}$ y $\mathbf{W}_{M_H}$ que modifica su estado según una observación codificada, $\vec{\mathbf{z}}_t$, y su estado en el instante anterior, $\vec{\mathbf{h}}_t$. |
| $d_m$: | dimensión del estado de la memoria del agente. |
| $\rho_1$: | radio espectral de la matriz $\mathbf{W}_{M_H}$. |
| $\vec{\mathbf{x}}$: | concatenación del código $\vec{\mathbf{z}}$ y del estado $\vec{\mathbf{h}}$. |

$\phi(\vec{\mathbf{x}}_t, \mathbf{a}_t; \mathbf{W}_\phi)$: modelo social. Red neuronal parametrizada por $\mathbf{W}_\phi$ que recibe una observación comprimida, el estado de la memoria y la acción en el instante presente y predice la acción de otros agentes en el instante siguiente.

$\mathbf{a}_{-i,t+1}$: acción conjunta de la población exceptuando el $i$-ésimo agente en el instante siguiente.

$J_\phi$: función objetivo a minimizar en el entrenamiento del modelo social.

$\mathcal{T}$: trayectoria de observaciones codificadas, estados y acciones, $(\vec{x}_t, a_t, a_{-i,t+1})$.

$\tau$: tupla de observaciones codificadas, estados y acciones $(\vec{x}_t, a_t, a_{-i,t+1})$.

$\vec{\mathbf{z}}_{a_{-i}}$: código generado a partir de $\vec{\mathbf{z}}$ que contiene únicamente la información concerniente a otros agentes en la observación.

$C'(\vec{\mathbf{z}}; \mathbf{W}_{C'})$: codificador auxiliar. Red neuronal parametrizada por $\mathbf{W}_{C'}$ que recibe una observación codificada, $\vec{\mathbf{z}}$, y genera el código, $\vec{\mathbf{z}}_{a_{-i}}$.

$D'(\vec{\mathbf{z}}_{a_{-i}}; \mathbf{W}_{D'})$: decodificador del codificador auxiliar. Red neuronal parametrizada por $\mathbf{W}_{D'}$ que recibe el código $\vec{\mathbf{z}}_{a_{-i}}$ y define para cada píxel una distribución de probabilidad de pertenencia a una de las $n_{c'} = 4$ clases relacionadas a otros agentes en el juego de la cosecha.

$n_{c'}$: número de clases relacionadas a otros agentes en el juego de la cosecha (agente, otro agente, mira del agente, rayo, u otro).

$K$: función que asigna la clase $c_7$ ="fondo" a cualquier píxel no relacionado a agentes.

$J_{C',D'}$: función objetivo a minimizar en el entrenamiento del codificador auxiliar.

$\mathbf{C}'$: matriz cuyas $n_{c'}$ columnas de la $i$-ésima fila corresponden a las probabilidades de pertenencia a cada clase relacionada a agentes para el $i$-ésimo píxel.

$\vec{z}_{a_{-i},t+2}$: valor que toma la variable aleatoria $\tilde{\mathbf{z}}_{a_{-i}}$ en el instante $t+2$.

| | |
|---|---|
| $\mathcal{T}'$: | trayectoria de observaciones codificadas, estados y acciones codificadas, $(\vec{z}_t, a_t, \vec{z}_{a_{-i},t+2})$. |
| $\tau'$: | tupla de observaciones codificadas, estados y acciones codificadas, $(\vec{z}_t, a_t, \vec{z}_{a_{-i},t+2})$. |
| $\hat{V}(\vec{\mathbf{x}}; \mathbf{W}_V)$: | red neuronal parametrizado por $\mathbf{W}_V$ que implementa la función de valor del controlador. |
| $\pi(\mathbf{a}, \vec{\mathbf{x}}; \mathbf{W}_\pi)$: | red neuronal parametrizada por $\mathbf{W}_\pi$ que implementa la política estocástica principal del controlador. |
| $\pi(\mathbf{a}, \vec{\mathbf{x}}; \mathbf{W}_\beta)$: | red neuronal parametrizada por $\mathbf{W}_\beta$ que implementa la política estocástica de referencia del controlador. |
| $\rho(\vec{\mathbf{x}}, \mathbf{a})$: | razón entre la política principal y la política de referencia: $\frac{\pi(\mathbf{a}, \vec{\mathbf{x}}; \mathbf{W}_\pi)}{\pi(\mathbf{a}, \vec{\mathbf{x}}; \mathbf{W}_\beta)}$. |
| $\epsilon$ : | híperparámetro del algoritmo PPO que controla la variación máxima permitida de $\rho(\vec{\mathbf{z}}, \mathbf{a})$ en una iteración. |
| $clip(i, a, b)$ : | función que acota el valor de $i$ al intervalo $[a, b]$. |
| $\hat{A}(\vec{\mathbf{x}}, \mathbf{a})$ : | estimado de la función de ventaja del controlador. |
| $G(\vec{\mathbf{x}}; \pi)$ : | ganancia obtenida a partir del instante $t$ siguiendo la política $\pi$ del controlador. |
| $w_H$ : | parámetro que pondera la entropía de la política en la función objetivo del controlador. Controla el nivel de exploración. |
| $w_F$ : | parámetro que pondera el estimado de la información mutua entre las acciones de los agentes en la función objetivo del controlador. |

**Sección 3.2**

$T_{offline}$: número de instantes de tiempo durante los que se ejecuta el sistema multi-agente de políticas uniformes para el muestreo de $\mathcal{O}$.

$t_{max}$: número de instantes de tiempo máximos de interacción con el entorno.

**Sección 3.3**

$r_{i,t}$: recompensa obtenida en el instante $t$ por el $i$-ésimo agente.

$\mathbf{O}_{i,t}$: observación obtenida en el instante $t$ por el $i$-ésimo agente.

$G_i$: ganancia no descontada obtenida por el $i$-ésimo agente en un episodio.

$U$: índice de utilidad del sistema multi-agente.

$E$: índice de equidad del sistema multi-agente.

$S$: índice de sostenibilidad del sistema multi-agente.

$P$: índice de paz del sistema multi-agente.

$\mathbf{O}_{TO}$: observación que recibe un agente en tiempo fuera.

$\bar{I}$: estimado de la información mutua promedio.

$\bar{H}$: entropía promedio.

*Una mente en permanente desarrollo solo puede tener un propósito*

*- cambiar la naturaleza de la Naturaleza.*

*Definitely Maybe, Arkady y Boris Strugatski*

# Capítulo 1

# Introducción

## 1.1. Planteamiento del problema

La última década ha visto un renovado interés en los algoritmos basados en *redes neuronales artificiales*, suscitado por los grandes avances en la aplicación de estos a problemas complejos, como visión artificial [1], traducción automática [2], síntesis y reconocimiento de voz [3], sistemas automáticos de juegos complejos [4], entre otros. Estos logros se han debido a técnicas enmarcadas en el denominado *aprendizaje profundo*[1], que busca extender las capacidades del aprendizaje de máquina al tipo de datos de alta dimensión y rica estructura que se encuentran en el mundo real [5], como imágenes naturales, señales de voz, o textos con gran variedad de palabras y símbolos.

La mayoría de estos avances se ha dado para el caso en que hay un solo aprendiz, o *agente*, y problemas que podrían beneficiarse de múltiples aprendices que actúen en conjunto han sido comparativamente poco explorados a la luz de las técnicas recientes [6]. A estos sistemas en que múltiples agentes interactúan mediante percepción y acción con su entorno y entre sí, se les denomina *sistemas multi-agente*, y es un concepto que abarca desde estructuras de la complejidad de la sociedad humana hasta sistemas propios de la ingeniería, como las redes de comunicaciones [7].

[1] Conjunto de técnicas basadas en redes neuronales de múltiples capas ocultas, denominadas *redes neuronales profundas*. Usualmente el término hace referencia a redes neuronales convolucionales o recurrentes (Consultar referencia [5]).

Desde el punto de vista de la ingeniería, los sistemas multi-agente cobran interés en problemas complejos que deben resolverse de forma distribuida, bien sea porque una solución centralizada no es posible, o porque se quiere hacer un uso eficiente de los recursos distribuidos [7]. En este trabajo se consideran problemas cuya solución óptima requiere que los agentes coordinen para maximizar una misma función objetivo que depende de las acciones conjuntas de la población, aún cuando sus objetivos individuales son divergentes, por lo que la cooperación será un fenómeno emergente. A estos problemas que presentan un conflicto entre la racionalidad individual y la racionalidad grupal, se les denomina *dilemas sociales* [8]. Un ejemplo práctico de tal situación puede presentarse en las redes móbiles ad hoc, en las que pueden haber nodos *egoístas* que usen los recursos de otros nodos para enviar sus paquetes pero que no prestan sus recursos a otros, aún cuando la eficiencia global de la red se beneficia de la cooperación entre nodos [9].

En [10], los autores identifican dos posibles enfoques al problema del entrenamiento de sistemas multi-agente: *aprendizaje de equipo*, que equivale al entrenamiento de un solo aprendiz que en lugar de aprender comportamientos individuales aprende comportamientos grupales, y *aprendizaje concurrente* en que a cada agente corresponde un proceso de aprendizaje. El aprendizaje de equipo permite la aplicación directa de las técnicas de aprendizaje para el caso de un solo agente, pero tiene problemas de escalabilidad con el número de agentes dado que su espacio de búsqueda es el espacio conjunto[2], y requiere además la centralización del aprendizaje, lo cual no es práctico en casos donde los recursos son inherentemente distribuidos y/o la comunicación entre agentes es costosa. El aprendizaje concurrente simplifica el espacio de búsqueda al proyectarlo en $N$ espacios más pequeños, para el caso de $N$ agentes, sin embargo la presencia de múltiples aprendices concurrentes hace que el problema sea no estacionario, lo que es una violación de la condición de estacionariedad asumida por la mayoría de algoritmos de aprendizaje estadístico [11].

Los desafíos asociados al aprendizaje en sistemas multi-agente cobran particular

[2] Para $N$ agentes, sea $\mathcal{S}_i$ el espacio de búsqueda del agente $i$, el espacio conjunto será: $\mathcal{S}_1 \times ... \times \mathcal{S}_N$.

importancia cuando los agentes son redes neuronales profundas. El problema de la escalabilidad se acentúa dado que el espacio de búsqueda[3] aún para el caso de un solo agente es de alta dimensión. Además, las técnicas tradicionales para el entrenamiento de agentes pueden divergir cuando se usan aproximadores funcionales parametrizados no lineales [12], como lo son las redes neuronales, por lo que el añadir la no estacionariedad producto del aprendizaje concurrente podría dificultar aún más la convergencia.

Muchas de las técnicas desarrolladas para el entrenamiento de sistemas multi-agente compuestos de redes neuronales han recurrido a la centralización del aprendizaje para resolver los problemas de coordinación y no estacionariedad [6, 13–18], con las altas demandas de memoria y ancho de banda que esto implica al requerir almacenar y transmitir gran cantidad de datos de alta dimensión, y que hacen que sean difícilmente aplicables a problemas prácticos.

Inspirándose en trabajos desde la *teoría de sistemas complejos* que parecen sugerir la existencia de una fuerte relación entre la dinámica de la *información*[4] y la emergencia de fenómenos como la vida [20], la evolución [21], la inteligencia [22] y las dinámicas en las organizaciones humanas [23, 24], y en la propuesta dada en [25] de usar la *información mutua*, un índice de correlación no lineal, para cuantificar el concepto de *sinergia*, en este trabajo se propone diseñar un algoritmo de entrenamiento que maximice la información mutua entre las acciones de los agentes como un posible mecanismo para lograr la emergencia de cooperación. La estimación de la información mutua requeriría únicamente conocimiento de las acciones de los agentes, las cuales pueden ser extraídas de las observaciones que estos hacen del entorno, por lo que el algoritmo podría plantearse de forma descentralizada. Según lo anterior se plantea la pregunta de investigación:

*¿Cómo promover la emergencia de cooperación en sistemas multi-agente neuronales mediante la maximización de la información mutua entre agentes?*

[3] $\mathbb{R}^n$ para una red neuronal con $n$ pesos.
[4] En el sentido de Shannon [19].

El resolver esta pregunta podría ayudar a solventar la necesidad de algoritmos descentralizados para el entrenamiento de sistemas multi-agente cooperativos basados en redes neuronales profundas, lo que facilitaría la creación de soluciones distribuidas a problemas complejos que requieran el procesamiento de datos no estructurados de alta dimensión.

## 1.2. Objetivos

### 1.2.1. Objetivo general

Diseñar un sistema multi-agente neuronal entrenado en un dilema social mediante aprendizaje profundo por refuerzo, teniendo como criterio de entrenamiento la maximización de la información mutua entre las acciones de los agentes.

### 1.2.2. Objetivos específicos

- Implementar un entorno de entrenamiento para la evaluación del sistema multi-agente en un dilema social.

- Implementar un sistema multi-agente compuesto de redes neuronales profundas, y el algoritmo de aprendizaje por refuerzo considerando la maximización de la información mutua entre agentes.

- Definir criterios para la evaluación del desempeño del sistema multi-agente en el entorno de entrenamiento.

- Evaluar según los criterios propuestos el desempeño del sistema multi-agente entrenado al considerar la maximización de la información mutua entre las acciones de los agentes, respecto a un sistema entrenado sin esta consideración.

## 1.3. Solución Propuesta

A partir del planteamiento del problema expuesto en la sección 1.1, se propone el diseño de un sistema multi-agente neuronal cuyo proceso de aprendizaje esté guiado por un objetivo de maximización de la información mutua entre las acciones individuales de los agentes, pretendiendo así, como se plantea en la pregunta de investigación, promover la emergencia de cooperación en el sistema.

Dado que se desea un algoritmo descentralizado cada agente en el sistema ejecuta de forma independiente y en simultáneo con otros agentes su proceso de aprendizaje. El problema del aprendizaje de cada agente se plantea entonces como el siguiente problema de optimización:

$$\max_{\mathbf{W}}[I(\mathbf{W}), J(\mathbf{W})] \tag{1.1}$$

, donde $\mathbf{W}$ son los parámetros de las redes neuronales de las que se compone el agente y que determinan sus acciones, $I$ es la información mutua entre las acciones del agente y las acciones de otros agentes, y $J$ es una medida de desempeño del agente en el problema.

El problema de la ecuación 1.1 es un problema de optimización multi-objetivo, sin embargo en este trabajo se reduce a un problema de un solo objetivo combinando linealmente las dos funciones objetivo:

$$\max_{\mathbf{W}}\{w_I\, I(\mathbf{W}) + w_J\, J(\mathbf{W})\} \tag{1.2}$$

, donde $w_I$ y $w_J$ son constantes que ponderan la importancia de cada función.

Las funciones $J$ e $I$ tienen representaciones diferenciables según se describe en las secciones 2.2.3 y 2.5.1, respectivamente, por lo que el problema de la ecuación 1.2 es resuelto de forma aproximada usando un algoritmo de gradiente ascendente.

El sistema es evaluado según los índices que se proponen en la sección 3.3 en el juego de la cosecha, un dilema social que se describe en la sección 2.4.3, comparándolo con un sistema de referencia entrenado sin la maximización de la información mutua.

# Capítulo 2

# Marco de referencia

## 2.1. Definiciones

En esta sección se definen algunos conceptos que son usados frecuentemente en el presente trabajo y de los que no existe una definición formal, o bien, existen múltiples definiciones según el contexto en que se traten. Las definiciones dadas no pretenden ser unívocas, sino que buscan contextualizar al lector del marco conceptual en que se tratan estos conceptos.

#### Agente

En este trabajo se adopta la definición de agente dada en [26] (Definición 64):

*Un agente es una entidad distinguida contenida en un sistema estrictamente más grande con capacidad de percepción, acción, y orientada a objetivos*

El agente es una *entidad distinguida* del resto de entidades en el sistema al poseer las características de percepción, acción y orientación a objetivos. El sistema que contiene al agente es "más grande" en el sentido de que el agente no puede ser todo lo que existe. La *percepción* es la capacidad del agente de ser afectado por el sistema que lo contiene, la *acción* es la facultad de *causar* cambios en el sistema que lo contiene, y la *orientación a objetivos* es la característica según la cual el agente usa su capacidad

de percepción y acción en conjunto para alterar el entorno según algún criterio. En [26] se dan diversas definiciones formales de estos conceptos.

### Entorno

Según la definición de agente, se define el entorno como el sistema que contiene al agente.

### Aprendizaje

El concepto de aprendizaje aquí tratado es el del *aprendizaje de máquina* o *aprendizaje automático* en el que el *aprendiz*, el objeto del aprendizaje, es un sistema de cómputo artificial. Dado que los agentes tratados en este trabajo son sistemas de cómputo artificiales, el concepto de aprendizaje de máquina dado en [27] se altera ligeramente para indicar que el aprendiz es un agente:

*Un agente se dice que aprende de la experiencia $\mathcal{E}$ con respecto a algún conjunto de problemas $\mathcal{P}$ y una medida de desempeño $J$, si su desempeño en el conjunto de problemas $\mathcal{P}$, medido según $J$, mejora dada la experiencia $\mathcal{E}$*

En el contexto del aprendizaje de agentes se dice además que el aprendizaje es *online* si ocurre a medida que el agente interactúa con el entorno u *offline* si el aprendizaje no se da en simultáneo con el proceso de interacción y de recolección de experiencias.

### Cooperación

La cooperación en este trabajo se define como la:

*Característica de un conjunto de acciones coordinadas en un sistema de múltiples agentes que conllevan a aumentar el desempeño del sistema*

La coordinación de los acciones de los agentes significa que estas no son independientes, por lo que una posible forma de cuantificarla es con un índice de correlación.

Sea $J$ una medida del desempeño de un sistema de múltiples agentes y sea $I$ un índice de correlación estrictamente positivo entre las acciones de lo agentes, se define entonces un índice de cooperación, $c$, en el sistema de agentes, como el producto entre el índice de correlación de las acciones de los agentes y su desempeño:

$$c = I \cdot J \tag{2.1}$$

El índice $c$ será alto para acciones altamente correlacionadas que conlleven a alto desempeño, y será igual a cero para acciones independientes aún cuando conlleven a un alto desempeño.

## 2.2. Aprendizaje por refuerzo

El aprendizaje por refuerzo es un modelo computacional del aprendizaje por interacción, fuertemente inspirado en la teoría de psicología conductivista [28]. Este consiste de un agente que interactúa con su entorno, y mediante una conexión sensorial con este, observa el efecto de sus acciones y aprende a modificar su comportamiento en respuesta a los estímulos positivos o negativos que recibe.

A diferencia del aprendizaje supervisado, en el aprendizaje por refuerzo al aprendiz no se le dice qué acciones tomar, sino que este debe descubrir mediante prueba y error qué acciones conllevan a un estímulo positivo máximo a largo plazo, condicionadas al estado del entorno. La búsqueda por prueba y error y los estímulos como señales de supervisión retrasadas en el tiempo son las características distintivas del aprendizaje por refuerzo [29, 30].

### 2.2.1. El problema del aprendizaje por refuerzo

El modelo estándar del aprendizaje por refuerzo consiste de un *agente* conectado a un *entorno* dinámico mediante percepción y acción. El agente interactúa con el entorno en una secuencia de instantes discretos $t \in \{0, 1, 2, ..., L\}$ donde $L \in \mathbb{N}$ es el *horizonte*. A cada instante $t$ el agente recibe el *estado* del entorno $\mathbf{s}_t \in \mathcal{S}$, siendo $\mathcal{S}$ el conjunto

de todos los estados, y con base en este selecciona una *acción* $\mathbf{a}_t \in \mathcal{A}$, siendo $\mathcal{A}$ el conjunto de todas las acciones. La acción cambia el estado del entorno a un estado $\mathbf{s}_{t+1}$ según la función dinámica del entorno $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ que define el conjunto de probabilidades de transición entre estados condicionadas a la acción ejecutada. El valor de esta transición es comunicado al agente en el instante siguiente mediante una *recompensa* $\mathbf{r}_{t+1}$, dada por la función de recompensa $R : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathcal{R}$, siendo $\mathcal{R} \subseteq \mathbb{R}$ el conjunto de todas las posibles recompensas. La figura 2-1 ilustra la interacción entre agente y entorno.

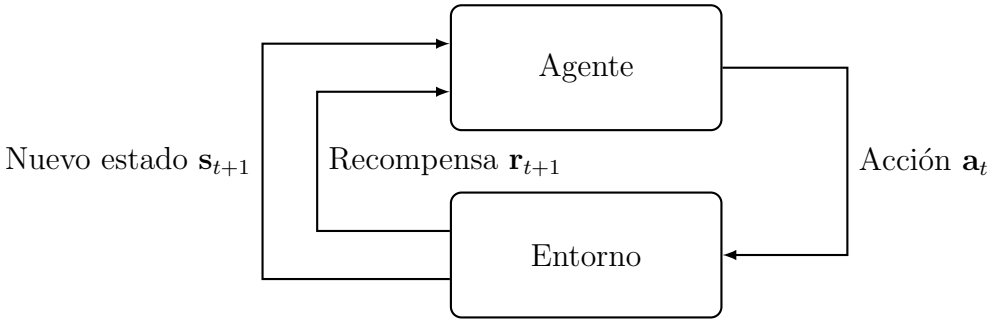


Figura 2-1: Esquema de la interacción entre agente y entorno. Traducción de la imagen mostrada en [29]

La elección de la acción a tomar por el agente se hace mediante una función $\pi : \mathcal{S} \to \mathcal{A}$, denominada *política*. Los algoritmos de aprendizaje por refuerzo buscan hallar la política óptima $\pi^*$ que maximice la ganancia $G$, definida como la recompensa total a un horizonte $L$ a partir de un estado inicial $\mathbf{s}_0$:

$$G(\mathbf{s}_0; \pi) = \sum_{t=0}^{L-1} \mathbf{r}_{t+1} \tag{2.2}$$

$$\pi^* = \underset{\pi}{\operatorname{argmax}}\, G(\mathbf{s}_0 = s; \pi) \ , \ \forall s \in \mathcal{S} \tag{2.3}$$

En tareas con horizonte $L = \infty$, para evitar la divergencia de la ganancia, se suele usar la ganancia descontada,

$$G(\mathbf{s}_0; \pi) = \sum_{t=0}^{L} \gamma^t \mathbf{r}_{t+1} \tag{2.4}$$

, donde $\gamma \in (0, 1)$ es el factor de descuento, que pondera el peso de las recompensas a corto plazo respecto a las recompensas a largo plazo. Puesto que $\gamma < 1$, las recompensas posteriores serán exponencialmente menos importantes que las inmediatas.

#### Procesos de decisión de Markov

Una forma frecuente de modelar un problema de aprendizaje por refuerzo es la del *proceso de decisión de Markov*. Este es un modelo frecuentemente usado en teoría de control estocástico [31] en el que se considera que el estado del sistema cambia aleatoriamente entre instantes de tiempo, y el futuro es condicionalmente independiente de estados pasados dado el estado presente. A esta característica se le llama la *propiedad de Markov*. Un proceso de decisión de Markov en aprendizaje por refuerzo está especificado por la tupla $(\mathcal{S}, \mathcal{A}, T, R)$.

### 2.2.2. Algoritmos de búsqueda indirecta de la política

Los algoritmos clásicos de aprendizaje por refuerzo hacen la suposición simplificadora de que el problema es un proceso de decisión de Markov, lo que implica que el estado presente es conocido y contiene toda la información necesaria para computar una acción óptima [32]. Esto permite la definición de una *función de valor*, $V$, definida como [1]:

$$V(s;\pi) = \mathbb{E}_{\mathbf{s}_{t+1}\sim T}\left[G(s;\pi)\right] = \sum_{s'\in\mathcal{S}} \mathbf{P}_{\mathbf{s}_{t+1}\,|\,\mathbf{s}_t,\mathbf{a}_t}(s' \mid s, \pi(s))(R(s') + G(s';\pi)) \tag{2.5}$$

La función $V(s;\pi)$ se denomina *función de valor de estado para la política* $\pi$, e indica el beneficio para el agente de estar en el estado $s$ si sigue la política $\pi$. De forma similar puede definirse una *función de valor de estado-acción* o *función de calidad*, que indica el beneficio de tomar una acción $a$ en el estado $s$ bajo la política $\pi$ [29]:

$$Q(s,a;\pi) = \sum_{s'\in\mathcal{S}} \mathbf{P}_{\mathbf{s}_{t+1}\,|\,\mathbf{s}_t,\mathbf{a}_t}(s' \mid s, a)(R(s') + G(s';\pi)) \tag{2.6}$$

[1] Asumiendo que $\pi$ y $R$ son determinísticas.

La política óptima $\pi^*$ dada la función $Q(s, a; \pi)$ puede encontrarse escogiendo "codiciosamente"[2] en cada estado:

$$\pi^*(s) = \underset{a}{\text{argmax}}\ Q(s, a; \pi) \tag{2.7}$$

Al conjunto de métodos que resuelven para la política óptima a través de las funciones de valor se les denomina *algoritmos de búsqueda indirecta*. Las técnicas de *programación dinámica* y los *métodos de Monte Carlo* aplicados a aprendizaje por refuerzo [29], *Q-learning* [33] y *SARSA* [34] son algunas de las técnicas pertenecientes a esta categoría.

### 2.2.3. Algoritmos de búsqueda directa de la política

Los *algoritmos de búsqueda directa* no hacen ninguna suposición respecto a la estructura del problema, por lo que pueden lidiar con entornos no Markovianos y no requieren hallar funciones de valor [32]. Estos algoritmos mantienen un estimado de la política como un aproximador funcional parametrizado que es evaluado directamente en el entorno según la recompensa obtenida. Los resultados de sucesivas evaluaciones brindan información para guiar la búsqueda de mejores soluciones.

#### Métodos basados en el gradiente de la Política

Una familia importante de algoritmos de búsqueda directa son los basados en el gradiente de la política. En estos, la política es un aproximador funcional parametrizado, $\pi(\mathbf{a}, \mathbf{s}; \omega_\pi)$, que se actualiza aproximadamente de forma proporcional al gradiente de $J_\pi$, una medida del desempeño de la política en función de la ganancia:

$$\Delta\omega_\pi \approx \alpha \nabla_{\mathbf{W}_\pi} J_\pi \tag{2.8}$$

, donde $\alpha \in (0, \infty)$ es la tasa de aprendizaje.

[2]Se usa el término "codicia" ya que en la literatura es común el término en inglés "greedy policy", haciendo referencia a escoger el argumento para valor máximo sin considerar otros factores.

Sea $\mathcal{E} = \{(s_0, a_0, r_1), ..., (s_{T-1}, a_{T-1}, r_L)\}$, una trayectoria de experiencias de longitud $L$ obtenidas online por el agente, del *teorema del gradiente de la política* para políticas estocásticas [35] se tiene que se puede obtener un estimado del gradiente de la función de desempeño de la forma:

$$\nabla_{\omega_\pi} J_\pi(\omega_\pi) = \frac{1}{|\mathcal{E}|} \sum_{(s,a,r)\in\mathcal{E}} [\nabla_{\omega_\pi} \ln(\pi(a, s; \omega_\pi))\, A(s, a)] \tag{2.9}$$

, donde $\pi(a, s; \omega_\pi) = p(\mathbf{a} = a \,|\, \mathbf{s} = s)$, es una política estocástica diferenciable respecto a $\omega_\pi$, y $A$ es la *función de ventaja*:

$$A(s, a) = Q(s, a; \pi) - V(s) \tag{2.10}$$

La función de ventaja usualmente es desconocida, en cuyo caso se usa en su lugar un estimado de esta usando los datos de la trayectoria $\mathcal{E}$,

$$\hat{A}(s_t, a_t) = r_{t+1} + \gamma r_{t+2} + ... + \gamma^{L-1} r_{t+L} + \gamma^L \hat{V}(s_{t+L}; \omega_V) - \hat{V}(s_t; \omega_V) \tag{2.11}$$

, donde $\hat{V}(s; \omega_V)$ es un aproximador de la función de valor entrenado para resolver de forma aproximada el problema de optimización de la ecuación 2.12,

$$\begin{aligned} &\min_{\omega_V} J_V \\ J_V(\omega_V) &= \sum_{(s_t, a_t, r_{t+1})\in\mathcal{E}} (G(s_t; \pi) - \hat{V}(s_t; \omega_V))^2 \end{aligned} \tag{2.12}$$

, siendo $G(s; \pi)$ la ganancia descontada de la ecuación 2.4 calculada para la trayectoria $\mathcal{E}$.

A los algoritmos donde se usa un aproximador para la función de valor que es usado para actualizar los parámetros de la política, se les denomina *algoritmos actor-crítico*. En un algoritmo actor crítico, el problema de optimización que define el proceso de entrenamiento de la política es entonces:

$$\max_{\omega_\pi} J_\pi$$
$$J_\pi(\omega_\pi) = \frac{1}{|\mathcal{E}|} \sum_{(s,a,r)\in\mathcal{E}} [\ln(\pi(a,s;\omega_\pi))\, \hat{A}(s,a)] \tag{2.13}$$

, donde a la política, $\pi$, se le llama *actor*, y al estimado de la función de ventaja, $\hat{A}$, *crítico*.

### 2.2.4. Aprendizaje profundo por refuerzo

La mayoría de problemas del mundo real involucran espacios de estado y de acción de alta dimensión, y en muchos casos continuos [36]. En tales problemas es apropiado usar las técnicas de aprendizaje profundo para complementar los algoritmos que requieren un estimado de las funciones de valor y/o de la política, dado que las redes neuronales se adaptan bien a entradas de alta-dimensión (series de tiempo, imágenes, etc.) y, en la práctica, no requieren de un incremento exponencial en datos al añadir dimensiones extra al espacio de estado o de acción [5,36,37]. Además, las redes neuronales pueden ser entrenadas incrementalmente y hacer uso de muestras adicionales obtenidas online [36]. Al conjunto de técnicas que involucran el uso de algoritmos de aprendizaje profundo en conjunto con algoritmos de aprendizaje por refuerzo, se les denomina *aprendizaje profundo por refuerzo*.

## 2.3. Sistemas multi-agente

Los sistemas multi-agente son aquellos en que múltiples agentes interactúan entre sí y con su entorno, usualmente bajo restricciones de observabilidad, al no poder conocer el estado entero del entorno, o los estados internos de los demás agentes [7]. En este trabajo se consideran sistemas de múltiples aprendices, es decir, donde los diversos agentes están aprendiendo simultáneamente, ignorando otros casos en que hay un único aprendiz y múltiples agentes de comportamiento fijo.

El problema del aprendizaje multi-agente es abordado generalmente desde el aprendizaje por refuerzo [10]. El aprendizaje en este caso es inherentemente más

complejo que en el caso de un solo agente dada la presencia de múltiples procesos dinámicos (los agentes cambiando en el tiempo a medida que aprenden) y las interacciones y dependencias entre estos.

### 2.3.1. El problema del aprendizaje por refuerzo multi-agente

#### Juegos de Markov

Los *juegos de Markov* son la extensión del concepto del proceso de decisión de Markov al caso multi-agente. Un juego de Markov está definido por la tupla $(\mathcal{S}, \mathcal{A}, T, R)$, donde se conserva la notación dada en la sección 2.2.1, salvo que el conjunto $\mathcal{A} = \mathcal{A}_1 \times \mathcal{A}_2 \times ... \times \mathcal{A}_N$, se redefine como el conjunto de todas las posibles acciones conjuntas, $\vec{\mathbf{a}} = [\mathbf{a}_1, ..., \mathbf{a}_N] \in \mathcal{A}$, siendo $\mathbf{a}_i \in \mathcal{A}_i$, la acción del $i$-ésimo agente, y $\mathcal{A}_i$ el conjunto de todas las posibles acciones del $i$-ésimo agente.

Dada la nueva definición de $\mathcal{A}$, las funciones de recompensa, $R$, y de transición, $T$, dependerán de las políticas de todos los agentes, y al ser estas variantes en el tiempo debido al proceso de aprendizaje, el problema descrito por la ecuación 2.3 se vuelve no estacionario [6].

## 2.4. Dilemas sociales

Los *dilemas sociales* son problemas en sistemas multi-agente en los que la racionalidad de los agentes puede darse en dos sentidos: la *racionalidad individual* que dicta a cada agente la estrategia más ventajosa según sus circunstancias, y la *racionalidad colectiva*, que dicta a la población entera de agentes en simultáneo la estrategia más ventajosa para la población. Un dilema social está definido por dos propiedades: **(a)** cada individuo recibe una utilidad mayor si actúa según su racionalidad individual, si no considera lo que otros individuos hagan, pero **(b)**, todos los individuos obtienen mayores utilidades si todos actúan según la racionalidad colectiva, que si todos actuaran según la racionalidad individual [8]. Este escenario, en que la ausencia de mecanismos de coordinación lleva inevitablemente a un sistema de agentes racionales

a una solución subóptima, plantea interesantes desafíos al desarrollo de un algoritmo descentralizado que lo resuelva.

### 2.4.1. Definición desde los juegos matriciales de suma no nula

La teoría de *juegos de suma no nula*[3] provee un marco teórico para formalizar los dilemas sociales mediante los *juegos matriciales*. Los juegos matriciales son sistemas de dos agentes, cada uno con un número finito de estrategias, $m_1$ y $m_2$, que se representan mediante una *matriz de utilidades*, $\mathbf{U} \in \mathbb{R}^{m_1 \times m_2}$, en que el elemento $\mathbf{U}_{i,j}$ es una tupla $(u_{1;\,i,j}, u_{2;\,i,j})$ , donde $u_{k;\,i,j}$ es la utilidad del agente $k$ si el agente 1 sigue la estrategia $i$, y el agente 2 sigue la estrategia $j$. La tabla 2.1 define un juego matricial con dos posibles acciones: cooperar ($c$), o no cooperar ($\neg c$), y cuatro posibles resultados: ganancia por cooperación mutua ($G_R$), castigo por comportamiento mutuamente egoísta ($G_P$), castigo por cooperar con un agente egoísta ($G_S$), y ganancia por explotar a un agente cooperativo ($G_T$).

| | $c$ | $\neg c$ |
|---|---|---|
| $c$ | $(G_R, G_R)$ | $(G_S, G_T)$ |
| $\neg c$ | $(G_T, G_S)\|$ | $(G_P, G_P)$ |

Tabla 2.1: Matriz de utilidades de un dilema social.

El juego matricial de la tabla 2.1 es entonces un dilema social si cumple las *desigualdades de los dilemas sociales* [38]:

1. $G_R > G_P$: recompensa mutua preferible a castigo mutuo.
2. $G_R > G_S$: recompensa mutua preferible a ser explotado.
3. $2G_R > G_T + G_S$: recompensa mutua preferible a alternar papel de explotador/explotado en un juego iterado.
4. $G_T > G_R$ ó $G_P > G_S$: explotar preferible a cooperar, o castigo mutuo preferible a ser explotado.

[3]Situaciones donde los jugadores pueden beneficiarse o perder al mismo tiempo, contrario a los juegos de suma cero en que la ganancia de unos implica la pérdida de otros, y viceversa.

Los juegos matriciales son un caso particular de los juegos de Markov, en que $N = 2$, $|\mathcal{S}| = 1$, y $\mathcal{A}_1 = \mathcal{A}_2 = \{c, \neg c\}$. Los posibles resultados se pueden expresar por medio de la función de valor como:

$$G_R(s) = V_1^{\pi^c,\pi^c}(s) = V_2^{\pi^c,\pi^c}(s) \tag{2.14}$$

$$G_P(s) = V_1^{\pi^{\neg c},\pi^{\neg c}}(s) = V_2^{\pi^{\neg c},\pi^{\neg c}}(s) \tag{2.15}$$

$$G_S(s) = V_1^{\pi^c,\pi^{\neg c}}(s) = V_2^{\pi^{\neg c},\pi^c}(s) \tag{2.16}$$

$$G_T(s) = V_1^{\pi^{\neg c},\pi^c}(s) = V_2^{\pi^c,\pi^{\neg c}}(s) \tag{2.17}$$

, donde $\pi^c$ es una política cooperativa, $\pi^{\neg c}$ es una política no cooperativa, y la función de valor de la ecuación 2.5 es modificada para el caso multi-agente:

$$V(s;\vec{\pi}) = \mathbb{E}_{\mathbf{s}_{t+1}\sim T}\left[G(s;\vec{\pi})\right] = \sum_{s'\in\mathcal{S}} \mathbf{P}_{\mathbf{s}_{t+1}\,|\,\mathbf{s}_t,\vec{\mathbf{a}}_t}(s' \mid s, \vec{\pi}(s))(R(s') + G(s';\vec{\pi})) \tag{2.18}$$

, siendo $\vec{\pi} = [\pi_1, \pi_2]$ la política conjunta.

Los juegos matriciales como herramienta para el estudio de dilemas sociales, aunque ampliamente usados, dejan de lado rasgos importantes de problemas del mundo real, al ignorar que los dilemas sociales frecuentemente son fenómenos que se extienden temporal y espacialmente, y que el cooperar o no cooperar no suelen ser acciones atómicas sino comportamientos o estrategias [39]. Ante esto, en [39] se propone el concepto de *dilema social secuencial*, que busca extender el concepto del dilema social a problemas temporal y espacialmente extendidos.

### 2.4.2. Dilemas sociales secuenciales

Un *dilema social secuencial* está definido por una tupla $(\mathcal{M}, \Pi^c, \Pi^{\neg c})$, donde $\Pi^c$ y $\Pi^{\neg c}$ son conjuntos disyuntos de políticas que se consideran cooperativas y no cooperativas, respectivamente, y $\mathcal{M}$ es un juego de Markov con $|\mathcal{S}| > 1$. Los posibles resultados ($G_R(s)$,$G_P(s)$,$G_S(s)$,$G_T(s)$) son inducidos por políticas, $\pi^c \in \Pi^c$ y $\pi^{\neg c} \in \Pi^{\neg c}$, según

las ecuaciones 2.14 a 2.18. Se dice entonces que un juego de Markov es un dilema social secuencial si su matriz de utilidad al organizar los posibles resultados como en la tabla 2.1 satisface las desigualdades de los dilemas sociales. En la siguiente sección se presenta el dilema social secuencial de *el juego de la cosecha*, que es el problema abordado en este trabajo.

### 2.4.3. El juego de la cosecha

El *juego de la cosecha*, propuesto en [40], es un dilema social secuencial en que un conjunto de agentes tienen como objetivo cosechar "manzanas"(recursos). La cantidad de manzanas es limitada, estas sólo crecen en determinadas áreas, y su tasa de crecimiento en un determinado punto en el espacio depende de la cantidad de manzanas en la vecindad de este: mientras más manzanas cercanas haya, a mayor velocidad crecerán. Si todas las manzanas en una región son cosechadas, ninguna volverá a crecer. Los agentes cuentan además con la posibilidad de dispararse entre sí un "rayo de tiempo-fuera". Cualquier agente que sea impactado por el rayo es penalizado con 25 instantes de tiempo fuera del juego.

El dilema yace en que la racionalidad individual dicta al individuo cosechar tan avidamente como pueda, sin embargo, la población como un todo se beneficia cuando los individuos se abstienen de cosechar, especialmente en situaciones donde muchos agentes cosechan en simultáneo en una misma región del espacio, pues es cuando es más probable que extingan el recurso. El rayo permite además la posibilidad de conflicto entre los agentes, que cuando se presenta, tiene como efecto reducir la población efectiva y por tanto la explotación del recurso. Intuitivamente una política no cooperativa será una política agresiva, en que los agentes frecuentemente intentan remover a sus competidores del juego para explotar egoístamente el recurso. En una política cooperativa en cambio, los agentes habrían de abstenerse ocasionalmente de explotar el recurso para compartirlo con otros. En [39] se demostró empíricamente que el juego de la cosecha es un dilema social secuencial para el caso de dos jugadores, y en [40] se generalizó para el caso de $N$ jugadores.

En la figura 2-2 se muestra un instante del juego de la cosecha. Un píxel en el

juego de la cosecha puede tomar $n_c = 7$ posibles valores: $c_1$ (manzana), $c_2$ (agente), $c_3$ (mira del agente), $c_4$ (otros agentes), $c_5$ (rayo de tiempo fuera), $c_6$ (muro), y $c_7$ (fondo). Las acciones $\vec{\mathbf{a}} \in \mathbb{R}^8$ son: paso arriba, paso abajo, paso a la derecha, paso a la izquierda, girar a la derecha, girar a la izquierda, disparar y quedarse quieto. La probabilidad de que aparezca una manzana en un píxel $x$, es una función del número de manzanas, $n_{mzn}$, en una vecindad de radio 2 centrada en $x$:

$$p(x = manzana \,|\, n_{mzn}) = \begin{cases} 0 & \text{Si } n_{mzn} = 0 \\ 0{,}01 & \text{Si } 0 < n_{mzn} \leq 2 \\ 0{,}05 & \text{Si } 2 < n_{mzn} \leq 4 \\ 0{,}1 & \text{Si } n_{mzn} > 4 \end{cases} \tag{2.19}$$

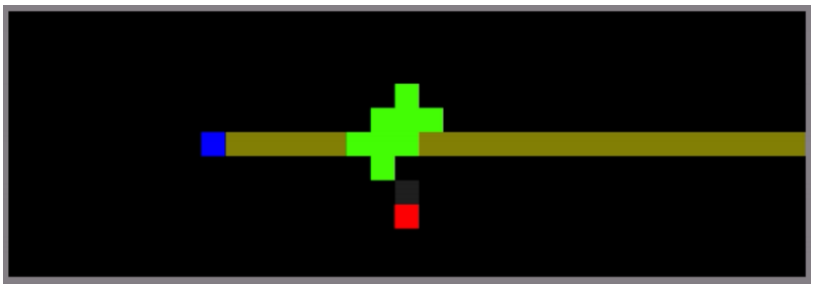

Figura 2-2: Un instante del juego de la cosecha. El agente (azul) dispara su rayo (amarillo verdoso) hacia el clúster de manzanas (verde), mientras otro agente (rojo) se aproxima desde el sur.

Formalmente, el juego de la cosecha es un juego de Markov parcialmente observable, $\mathcal{M} = (\mathcal{S}, \mathcal{O}, \mathcal{A}, T, R)$, donde $\mathcal{O} \subseteq \mathbb{R}^{d \times n_c}$ es el conjunto de todas las posibles observaciones, que es imagen de la función $O : \mathcal{S} \times \mathcal{X} \to \mathcal{O}$, donde $\mathcal{X} \subseteq \mathbb{Z}^2$ es el conjunto de todas las posibles posiciones del agente en el espacio. La función $O$ define entonces la visión $d$-dimensional que tiene cada agente del estado del entorno.

En cada instante la función de observación mapea para cada agente según el estado, $\mathbf{S}_t$, y su posición $(\mathbf{x},\mathbf{y})$, la observación que este recibe, $\mathbf{O}_t = O(\mathbf{S}_t, \mathbf{x}, \mathbf{y}) \in \mathbb{R}^{81 \times 7}$, que en el juego de la cosecha es una imagen de la vecindad de radio $r = 4$ centrada en el agente, en que cada a píxel corresponde un vector base canónica $\vec{b}^{(c_i)} \in \mathbb{R}^{n_c}$, si el píxel tiene el valor $c_i$. La observación es entonces una matriz de

$d = (2r+1)^2 = 81$ filas, considerando el píxel correspondiente al agente, y $n_c = 7$ columnas.

## 2.5. Información mutua

La *Información Mutua* es una cantidad adimensional que mide la reducción en la incertidumbre de una variable aleatoria dado el conocimiento de otra variable aleatoria. Para dos variables aleatorias discretas $\mathbf{x}$ y $\mathbf{y}$, con distribución conjunta $P_{\mathbf{xy}}(x,y)$, la información mutua entre ellas $I(\mathbf{x},\mathbf{y})$ es

$$I(\mathbf{x},\mathbf{y}) = \sum_{xy} P_{\mathbf{xy}}(x,y) \ln\left(\frac{P_{\mathbf{xy}}(x,y)}{P_{\mathbf{x}}(x)P_{\mathbf{y}}(y)}\right) \tag{2.20}$$

, donde $P_{\mathbf{x}}(x)$ y $P_{\mathbf{y}}(y)$ son las probabilidades marginales:

$$P_{\mathbf{x}}(x) = \sum_{y} P_{\mathbf{xy}}(x,y) \tag{2.21}$$

$$P_{\mathbf{y}}(y) = \sum_{x} P_{\mathbf{xy}}(x,y) \tag{2.22}$$

La información mutua tambien puede expresarse en términos de la entropía, una medida de la incertidumbre de una variable aleatoria, como

$$I(\mathbf{x},\mathbf{y}) = H(\mathbf{x}) - H(\mathbf{x}|\mathbf{y}) \tag{2.23}$$

, donde $H(\mathbf{x})$ es la entropía de $\mathbf{x}$:

$$H(\mathbf{x}) = -\sum_{x} P_{\mathbf{x}}(x) \ln(P_{\mathbf{x}}(x)) \tag{2.24}$$

, y $H(\mathbf{x}|\mathbf{y})$ es la entropía condicional:

$$H(\mathbf{x}|\mathbf{y}) = \sum_{y} P_{\mathbf{y}}(y) \left[ -\sum_{x} P_{\mathbf{x}|\mathbf{y}}(x|y) \ln(P_{\mathbf{x}|\mathbf{y}}(x|y)) \right] \tag{2.25}$$

La ecuación 2.23 expresa la información mutua como la cantidad de incertidumbre

en $\mathbf{y}$ menos la cantidad de incertidumbre en $\mathbf{y}$ una vez $\mathbf{x}$ es conocida, es decir, la cantidad de incertidumbre en $\mathbf{y}$ que se elimina al conocer $\mathbf{x}$ [19].

### 2.5.1. Estimador Neuronal de la Información Mutua

En [41] se propone un *Estimador Neuronal de la Información Mutua (MINE[4])* a partir de muestras de las variables aleatorias. El algoritmo usa la representación de la información mutua como una *Divergencia de Kullback-Leibler*, una métrica de distancia entre distribuciones de probabilidad definida como:

$$D_{KL}(P_{\mathbf{z}}||Q_{\mathbf{z}}) = \sum_{z} P_{\mathbf{z}}(z)\ln\left(\frac{P_{\mathbf{z}}(z)}{Q_{\mathbf{z}}(z)}\right) \tag{2.26}$$

Se puede observar de la ecuación 2.20 que la información mutua entre dos variables aleatorias es la divergencia $D_{KL}$ entre su distribución conjunta y el producto de las marginales

$$I(\mathbf{x},\mathbf{y}) = D_{KL}(P_{\mathbf{xy}}||P_{\mathbf{x}}P_{\mathbf{y}}) \tag{2.27}$$

La divergencia $D_{KL}$, según la *Representación de Donsker-Varadhan*, puede ser escrita como

$$D_{KL}(P_{\mathbf{z}}||Q_{\mathbf{z}}) = \sup_{F:\Omega\to\mathbb{R}} \left\{\mathbb{E}_{\mathbf{z}\sim P_{\mathbf{z}}}[F(z)] - \ln(\mathbb{E}_{\mathbf{z}\sim Q_{\mathbf{z}}}[e^{F(z)}])\right\} \tag{2.28}$$

, siendo $F : \Omega \to \mathbb{R}$ cualquier función que cuyo dominio es el espacio muestral, $\Omega$, y cuyo rango son los reales. Los autores parten de esa representación para definir una *medida neuronal de información*:

$$I(\mathbf{x},\mathbf{y}) = \sup_{\mathbf{W}_F} \left\{\mathbb{E}_{(\mathbf{x},\mathbf{y})\sim P_{\mathbf{xy}}}[F(x,y;\mathbf{W}_F)] - \ln(\mathbb{E}_{(\mathbf{x},\mathbf{y})\sim P_{\mathbf{x}}P_{\mathbf{y}}}[e^{F(x,y;\mathbf{W}_F)}])\right\} \tag{2.29}$$

, donde $F$ es implementada como una red neuronal diferenciable. La información mutua puede entonces ser aproximada resolviendo un problema de optimización en

[4]Mutual Information Neural Estimator.

el que se define una función $J_F$ a ser maximizada

$$\begin{aligned} &\max_{\mathbf{W}_F} J_F \\ J_F(\mathbf{W}_F) &= \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim P_{\mathbf{xy}}}[F(x,y;\mathbf{W}_F)] - \ln(\mathbb{E}_{(\mathbf{x},\mathbf{y})\sim P_{\mathbf{x}}P_{\mathbf{y}}}[e^{F(x,y;\mathbf{W}_F)}]) \end{aligned} \tag{2.30}$$

Dado que $F$ es una función diferenciable, el problema de la ecuación 2.30 puede ser resuelto de forma aproximada usando el algoritmo de retropropagación del error.

**Estimador basado en la Divergencia de Jensen-Shannon**

En [42] se propuso una modificación a MINE que reemplaza la divergencia $D_{KL}$ por la representación de la *Divergencia de Jensen-Shannon* propuesta en [43], resultando en la función objetivo a maximizar:

$$J_F(\mathbf{W}_F) = \ln(2{,}0) - \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim P_{\mathbf{xy}}}[-\zeta(-F(x,y;\mathbf{W}_F))] - \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim P_{\mathbf{x}}P_{\mathbf{y}}}[\zeta(F(x,y;\mathbf{W}_F))] \tag{2.31}$$

, donde $\zeta$, es la función *softplus*:

$$\zeta(x) = \ln(1+e^x) \tag{2.32}$$

Este estimador según se reporta en [42] mejora la convergencia del entrenamiento respecto al estimador de la ecuación 2.29. Aunque no aproxima el valor exacto de la información mutua, se comporta de forma similar al estimador de la ecuación 2.29, dado que ambos actúan como discriminadores cuyo objetivo es maximizar el valor esperado del logaritmo de la razón entre la distribución conjunta y el producto de las distribuciones marginales.

# Capítulo 3

# Diseño del sistema multi-agente neuronal

En este capítulo se presenta el diseño propuesto del sistema multi-agente neuronal que es entrenado en el juego de la cosecha. Dado que el objetivo es plantear un algoritmo descentralizado, en este trabajo se adopta la filosofía de diseño de la *ingeniería de sistemas complejos* [44], según la cual en lugar de diseñar el sistema parte por parte (agente por agente), se diseñan las reglas de interacción locales entre las partes de modo que el comportamiento deseado emerja de su auto-organización. Nuestro sistema de agentes es homogéneo, es decir, todos los agentes en el sistema son estructuralmente iguales, por lo que al especificar la estructura y reglas de un agente se está describiendo el diseño del sistema entero. En las siguientes secciones se presenta entonces tal diseño mediante la especificación de los elementos funcionales que definen los procesos de percepción, acción y aprendizaje del agente. Finalmente se describen los índices según los cuales se evaluará el desempeño del sistema.

## 3.1. Arquitectura del agente

El agente se compone de cuatro elementos: el *sensor*, la *memoria*, la *guía social*, y el *controlador*, tal como se ilustra en la figura 3-1. El *sensor* recibe las observaciones del entorno y las transforma a un código representativo de menor dimensión, redu-

ciendo por tanto el espacio de búsqueda. La *memoria* es un sistema dinámico que almacena información sobre la historia del estado del entorno. La *guía social* es el elemento encargado de guiar el proceso de aprendizaje de modo que, en presencia de otros agentes, el agente actúe coordinadamente con ellos. Esta consta de dos subcomponentes, el *modelo social*, que trata de predecir las acciones de otros agentes ante un determinado estado y acción del agente, y el *crítico social*, que mide la información mutua entre las acciones del agente y la de sus pares en el instante siguiente. Finalmente, el agente actúa según el *controlador*, que define la probabilidad de que el agente tome una determinada acción según el estímulo sensorial que reciba del sensor y el estado de la memoria.

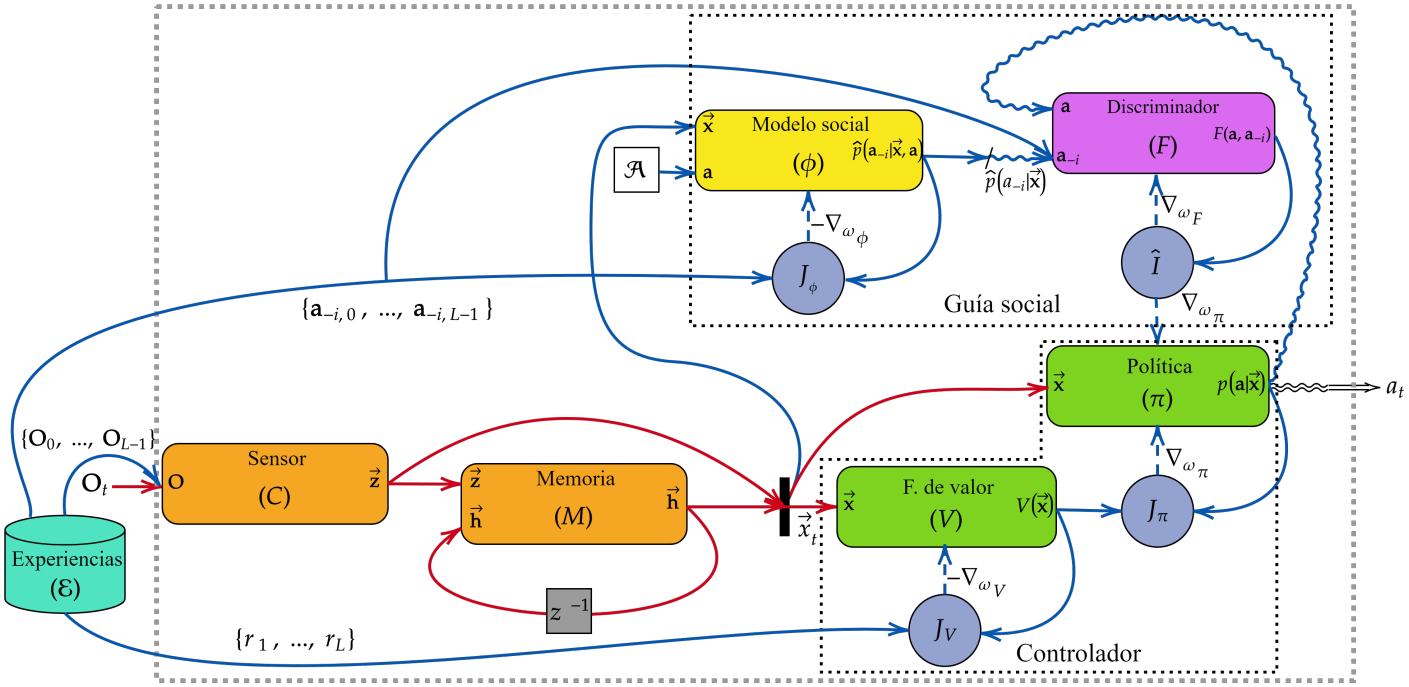


Figura 3-1: Arquitectura del agente. Cada red neuronal (bloques rectangulares) es sintonizada (líneas punteadas) mediante los gradientes de las funciones objetivo (círculos). Las líneas en color rojo corresponden al proceso de inferencia, mientras las de color azul corresponden al proceso de entrenamiento. Las líneas con rizado representan operaciones de muestreo de una variable aleatoria.

A continuación se describe cada elemento especificando la función que implementa la red neuronal, el problema de optimización según el cual se sintonizan sus

parámetros, y su algoritmo de entrenamiento. Todos los problemas de optimización se resuelven de forma aproximada usando el algoritmo de retropropagación del error.

### 3.1.1. Sensor

El sensor es una función, $C : \mathcal{O} \to \mathbb{R}^{d_z};\ \mathcal{O} \subseteq \mathbb{R}^{d \times n_c}$, donde $d_z < d$. Este se implementa como una red neuronal, $C(\mathbf{O}; \mathbf{W}_C)$. El sensor es entrenado offline como el componente codificador de un *autoencoder*.

#### Entrenamiento del autoencoder

Un autoencoder es un tipo de red neuronal diseñada para aprender una codificación de un conjunto de datos, típicamente para reducir su dimensión [5]. Se compone de dos partes, el codificador que genera un código, $\vec{\mathbf{z}} = C(\mathbf{O}; \mathbf{W}_C)$, y el decodificador, una red neuronal, $D(\vec{\mathbf{z}}; \mathbf{W}_D)$, encargada de generar una reconstrucción de la entrada, $\hat{\mathbf{O}} \in \mathcal{O}$, a partir del código $\vec{\mathbf{z}}$. Dado que como se describió en la sección 2.4.3 las observaciones en el juego de la cosecha consisten de imágenes cuyos píxeles solo pueden tomar $n_c = 7$ valores discretos[1], se decidió entrenar el decodificador como un clasificador multi-clase, donde cada clase corresponde a cada posible valor de un píxel. La elección de entrenar el decodificador como un clasifificador, es decir un sistema de salidas discretas (categóricas), permite que las reconstrucciones sean más robustas a ruido y errores.

La función $D : \mathbb{R}^{d_z} \to [0, 1]^{d \times n_c}$, define entonces una distribución de probabilidad de pertenencia a cada una de las $n_c$ clases para cada píxel de la entrada reconstruida. En la imagen reconstruida el $i$-ésimo píxel, $\hat{\mathbf{O}}_{i,:}$, toma el valor más probable según $D$:

$$\hat{\mathbf{O}}_{i,:} = \vec{b}^{(k)} \quad ; \quad k = \underset{j}{\operatorname{argmax}}\ \mathbf{C}_{:,j} \tag{3.1}$$

, donde $\vec{b}^{(k)} \in \mathbb{R}^{n_c}$ es la $k$-ésima base canónica, y $\mathbf{C} = D(\vec{\mathbf{z}}; \mathbf{W}_D)$, es la matriz de probabilidades de pertenencia para cada píxel.

[1] $c_1$ (manzana), $c_2$ (agente), $c_3$ (mira del agente), $c_4$ (otros agentes), $c_5$ (rayo de tiempo fuera), $c_6$ (muro), y $c_7$ (fondo).

El proceso de entrenamiento del autoencoder se formula como el siguiente problema de optimización:

$$\min_{\mathbf{W}_C, \mathbf{W}_D} J_{C,D}\left(D(C(\mathbf{O}; \mathbf{W}_C); \mathbf{W}_D),\, \mathbf{O}\right); \quad \forall \mathbf{O} \in \mathcal{D} \tag{3.2}$$

, donde $J_{C,D}$ es una función de error, y $\mathcal{D}$ es un conjunto de observaciones muestreadas uniformemente de $\mathcal{O}$. La función de entropía cruzada promedio es usada como función de error del autoencoder:

$$J_{C,D} = -\frac{1}{|\mathcal{D}|} \sum_{\mathbf{O} \in \mathcal{D}} \sum_{i=1}^{d} \sum_{j=1}^{n_c} \mathbf{O}_{i,j} \ln(\mathbf{C}_{i,j}) \tag{3.3}$$

, donde $\mathbf{C} = D(C(\mathbf{O}; \mathbf{W}_C); \mathbf{W}_D)$.

El algoritmo 1 describe el proceso de entrenamiento del sensor.

**Algoritmo 1** Entrenamiento del sensor

**procedure** ENTRENAR_SENSOR($n_{iters}$: número de iteraciones del entrenamiento, $b$: tamaño de los lotes de entrenamiento, $\alpha_C$: tasa de aprendizaje del sensor)
  Inicializar $\mathbf{W}_C$ y $\mathbf{W}_D$
  Muestrear uniformemente $\mathcal{O}$ para obtener conjunto de datos $\mathcal{D}$
  **for** $n_{iters}$ **do**
    **for** FLOOR($|\mathcal{D}| \div b$) **do**
      Muestrear $b$ elementos uniformemente de $\mathcal{D}$ para obtener lote $\mathcal{B}$
      Calcular:

$$J_{C,D} = -\frac{1}{b} \sum_{\mathbf{O} \in \mathcal{B}} \sum_{i=1}^{d} \sum_{j=1}^{n_c} \mathbf{O}_{i,j} \ln(\mathbf{C}_{i,j})$$

      Actualizar parámetros del autoencoder:

$$\mathbf{W}_C = \mathbf{W}_C - \alpha_C \nabla_{\mathbf{W}_C} J_{C,D}$$

$$\mathbf{W}_D = \mathbf{W}_D - \alpha_C \nabla_{\mathbf{W}_D} J_{C,D}$$

    **end for**
  **end for**
  **return** $\mathbf{W}_C, \mathcal{D}$
**end procedure**

### 3.1.2. Memoria

La memoria del agente es un *reservoir*, siguiendo la terminología del *reservoir computing* [45], un sistema dinámico de alta dimensión cuyo estado $\vec{\mathbf{h}}$ varía en el tiempo según el código $\vec{\mathbf{z}}$ emitido por el sensor, que recibe como entrada, y de su estado en instantes anteriores:

$$\vec{\mathbf{h}}_t = M(\vec{\mathbf{z}}_t, \vec{\mathbf{h}}_{t-1}) \tag{3.4}$$

, donde $\vec{\mathbf{h}}_t \in \mathbb{R}^{d_M}$, con $d_M \gg d_z$.

La función $M$ se implementa como una red neuronal recurrente[2],

$$\vec{\mathbf{h}}_t = M(\vec{\mathbf{z}}_t, \vec{\mathbf{h}}_{t-1}; \mathbf{W}_{M_I}, \mathbf{W}_{M_H}) = \tanh(\mathbf{W}_{M_I}\, \vec{\mathbf{z}}_t + \mathbf{W}_{M_H}\, \vec{\mathbf{h}}_t) \tag{3.5}$$

, donde las matrices de pesos $\mathbf{W}_{M_I}$ y $\mathbf{W}_{M_H}$ son inicializadas aleatoriamente y se mantienen constantes durante el entrenamiento del agente. La elección de una red reservoir en lugar de una red neuronal recurrente sintonizable se hizo considerando los altos costos computacionales del algoritmo de *retropropagación del error en el tiempo* necesario para entrenarlas [46].

La matriz $\mathbf{W}_{M_I}$ es inicializada muestreando de una distribución Gaussiana estándar, y La matriz $\mathbf{W}_{M_H}$ es inicializada según la ecuación 3.6,

$$\mathbf{W}_{M_H} = \frac{\rho_1}{\rho_0(\mathbf{W}_{M_H,0})}\, \mathbf{W}_{M_H,0} \tag{3.6}$$

, donde $\mathbf{W}_{M_H,0}$ es una matriz muestreada de una distribución Gaussiana estándar, $\rho_0(\mathbf{W}_{M_H,0}) = \text{máx}(|\lambda|)$ es el *radio espectral* de $\mathbf{W}_{M_H,0}$ , siendo $\lambda$ sus valores propios, y $\rho_1$ es el valor deseado del radio espectral. El radio espectral de la matriz de pesos del reservoir debe ser menor a la unidad, $\rho_1 < 1$, para que el sistema sea estable en el sentido de Lyapunov [47], y su magnitud es directamente proporcional a la capacidad de memoria del sistema [48].

La secuencia de observaciones que entran al agente inducen una variedad de res-

[2]Para información de redes neuronales recurrentes consúltese la referencia [5].

puestas no lineales en cada neurona del reservoir. Estas respuestas representan patrones temporales del estado del entorno, y pueden ser usadas por el agente para compensar por la observabilidad parcial [49]. La concatenación del código emitido por el sensor y el estado del reservoir, $\vec{\mathbf{x}}_t = [\vec{\mathbf{z}}_t, \vec{\mathbf{h}}_t]^T$ es usada como entrada en otros componentes del agente.

### 3.1.3. Guía social

#### Modelo social

El modelo social del $i$-ésimo agente es una función $\phi : \mathbb{R}^{(d_z+d_M)} \times \mathcal{A}_i \rightarrow [0,1]^{|\mathcal{A}_{-i}|}$, donde $\mathcal{A}_{-i}$ es el conjunto de todas las posibles acciones conjuntas de la población menos el agente $i$-ésimo. La función $\phi$ es implementada como una red neuronal, $\phi(\vec{\mathbf{x}}_t, \mathbf{a}_t; \mathbf{W}_\phi)$, que estima $p(\mathbf{a}_{-i,t+1}|\vec{\mathbf{x}}_t, \mathbf{a}_t)$, donde $\mathbf{a}_{-i,t+1}$ es la acción conjunta de la población exceptuando el $i$-ésimo agente, y que se usa para predecir el efecto de las acciones del agente en las acciones de otros agentes dada la observación presente codificada y el estado de la memoria. El entrenamiento del modelo social consiste en resolver el problema de optimización de la ecuación 3.7,

$$\min_{\mathbf{W}_\phi} J_\phi \left(\phi(\vec{x}_t, a_t; \mathbf{W}_\phi), a_{-i,t+1}\right); \quad \forall(\vec{x}_t, a_t, a_{-i,t+1}) \in \mathcal{T} \tag{3.7}$$

donde $\mathcal{T} = \{(\vec{x}_0, a_0, a_{-i,1}), ..., (\vec{x}_{L-1}, a_{L-1}, a_{-i,L})\}$ es una trayectoria de observaciones codificadas, estados y acciones de longitud $L$ obtenidas online por el agente, y $J_\phi$ es la función de entropía cruzada promedio:

$$J_\phi = -\frac{1}{|\mathcal{T}|} \sum_{\tau \in \mathcal{T}} \sum_{i=1}^{|\mathcal{A}_{-i}|} \vec{b}^{(a_{-i,t+1})} \ln(\hat{\vec{y}}_i) \tag{3.8}$$

, siendo $\vec{b}^{(k)} \in \mathbb{R}^{|\mathcal{A}_{-i}|}$ es la $k$-ésima base canónica, $\tau = (\vec{x}_t, a_t, a_{-i,t+1})$ y $\hat{\vec{y}} = \phi(\vec{x_t}, a_t; \mathbf{W}_\phi)$.

El algoritmo 2 describe el proceso de entrenamiento del modelo social.

**Algoritmo 2** Entrenamiento del modelo social

**procedure** ENTRENAR_MODELO_SOCIAL($n_{iters}$: número de iteraciones del entrenamiento, $\mathcal{T}$: trayectoria observaciones, estados y acciones, $\alpha_\phi$: tasa de aprendizaje del modelo social)
  **for** $n_{iters}$ **do**
    Calcular $J_\phi$ (Ecuación 3.8)
    Actualizar parámetros del modelo:

$$\mathbf{W}_\phi = \mathbf{W}_\phi - \alpha_\phi \nabla_{\mathbf{W}_\phi} J_\phi$$

  **end for**
  **return** $\mathbf{W}_\phi$
**end procedure**

### Crítico social

El estimador de la ecuación 2.31 es usado como índice de la información mutua entre las acciones del agente en el instante presente, $\mathbf{a}_t$, y las acciones de otros agentes en el instante siguiente, $\mathbf{a}_{-i,t+1}$, condicionada al código de la observación hecha por el agente en el instante presente y al estado de la memoria, $\vec{\mathbf{x}}_t$. El índice de la información mutua se obtiene resolviendo el siguiente problema de optimización:

$$\begin{aligned}\max_{\mathbf{W}_F} \Big\{ \ln(2{,}0) - \mathbb{E}_{(\mathbf{a}_t, \mathbf{a}_{-i,t+1}) \sim P_{\mathbf{a}_t \mathbf{a}_{-i,t+1} | \vec{\mathbf{x}}_t}} [-\zeta(-F(a_t, a_{-i,t+1}; \mathbf{W}_F))] - ... \\ \mathbb{E}_{(\mathbf{a_t}, \mathbf{a}_{-i,t+1}) \sim P_{\mathbf{a_t} | \vec{\mathbf{x}}_t} P_{\mathbf{a}_{-i,t+1} | \vec{\mathbf{x}}_t}} [\zeta(F(a_t, a_{-i,t+1}; \mathbf{W}_F))] \Big\}\end{aligned} \tag{3.9}$$

Dada la alta dimensión del espacio muestral de $\vec{\mathbf{x}}_t$ no es factible computar los valores esperados de la ecuación 3.9 de forma exacta, por lo que estos son estimados como promedios sobre muestras de las distribuciones. Las muestras de la distribución conjunta, $P_{\mathbf{a}_t \mathbf{a}_{-i,t+1} | \vec{\mathbf{x}}_t}$, se obtienen de la trayectoria $\mathcal{T}$. La distribución marginal $P_{\mathbf{a}_t | \vec{\mathbf{x}}_t}$ es la política del agente. La distribución marginal $P_{\mathbf{a}_{-i,t+1} | \vec{\mathbf{x}}_t}$ es desconocida por el agente, pero pueden obtenerse muestras de una distribución aproximada al marginalizar $\mathbf{a}_t$ del modelo social:

$$p(\mathbf{a}_{-i,t+1} | \vec{\mathbf{x}}_t) = \sum_{\mathbf{a}_t \in \mathcal{A}_i} p(\mathbf{a}_{-i,t+1} | \vec{\mathbf{x}}_t, \mathbf{a}_t) \approx \sum_{a_t \in \mathcal{A}_i} \phi(\vec{\mathbf{x}}_t, a_t) \tag{3.10}$$

La función objetivo a maximizar es entonces:

$$J_F = \ln(2{,}0) + \frac{1}{|\mathcal{T}|}\sum_{\tau\in\mathcal{T}} \zeta(-F(\pi(\vec{x}_t, a_t; \mathbf{W}_\pi), \vec{a}^{(a_{-i,t+1})}; \mathbf{W}_F)) - ... \\ \frac{1}{|\mathcal{T}|}\sum_{\tau\in\mathcal{T}} \zeta\left(F\left(\pi(\vec{x}_t, a_t; \mathbf{W}_\pi), \sum_{a_t\in\mathcal{A}_i} \phi(\vec{x}_t, a_t); \mathbf{W}_F\right)\right) \tag{3.11}$$

El crítico social es entrenado conjuntamente con la política de control, por lo que su algoritmo de entrenamiento se describe en la sección 3.1.5.

### 3.1.4. Guía social modificada para el caso de más de dos agentes

El modelo social y el crítico social descritos en la sección anterior escalan pobremente a medida que aumenta la cantidad de agentes. El rango de la función $\phi$ crece a razón de $|\mathcal{A}_{-i}| = |\mathcal{A}_i|^{N-1}$ con el tamaño de la población[3], $N$, al igual que el dominio del discriminador del crítico social, $\Omega$, que crece a razón de $|\Omega| = |\mathcal{A}_i| + |\mathcal{A}_i|^{N-1}$. Además, cada agente debería ser visualmente único para ser distinguible uno de otro en las observaciones del estado, y el algoritmo no funcionaría para poblaciones de composición variable en el tiempo.

Para solucionar estos problemas de escalabilidad, la guía social es modificada partiendo del hecho de que las acciones que realicen todos los agentes en el campo de visión de un determinado agente serán en su mayoría apreciables en la observación que haga el agente en el instante siguiente. Por tanto, maximizar la información mutua entre los píxeles correspondientes a otros agentes[4] dos instantes en el futuro, y las acciones del agente, corresponderá de forma aproximada a maximizar la información mutua entre las acciones de todos los agentes visibles por el agente, y sus acciones en el instante siguiente. Los componentes de la guía social modificada se ilustran en la figura 3-2

[3]Por la regla del producto: $|\mathcal{A}_1 \times ... \times \mathcal{A}_{N-1}| = |\mathcal{A}_1| \times ... \times |\mathcal{A}_{N-1}|$

[4]agente, otro agente, mira, y rayo.

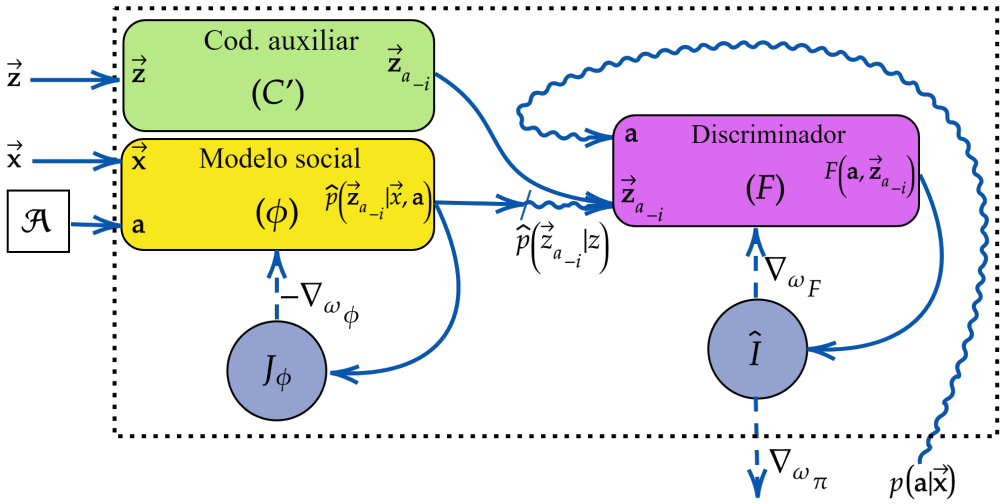


Figura 3-2: Arquitectura de la guía social modificada para el caso de más de dos agentes.

### Codificador auxiliar

La información de los píxeles correspondientes a otros agentes y sus acciones se encuentra contenida en el código $\vec{\mathbf{z}}$ emitido por el sensor, por lo que para extraerla se entrena offline un codificador auxiliar cuya función es obtener un código $\vec{\mathbf{z}}_{a_{-i}}$ a partir del código $\vec{\mathbf{z}}$, que contenga únicamente la información concerniente a píxeles de agentes. Este es entrenado con el procedimiento descrito en la sección 3.1.1 para el sensor.

En este caso el codificador es una función $C' : \mathbb{R}^{d_z} \to \mathbb{R}^{d_{z'}}$, siendo $d_{z'}$ la dimensión del código $\vec{\mathbf{z}}_{a_{-i}}$. El decodificador, $D' : \mathbb{R}^{d_{z'}} \to [0,1]^{d_x \times n_{c'}}$, entrenado como un clasificador multi-clase, define la probabilidad para cada píxel de pertenecer a una de $n_{c'} = 5$ posibles clases: $c_2$ (agente), $c_3$ (mira del agente), $c_4$ (otro agente), $c_5$ (rayo de tiempo fuera), o $c_7$ (fondo)[5]. El problema de entrenamiento del codificador auxiliar queda descrito por las ecuaciones 3.12 y 3.14.

$$\min_{\mathbf{W}_{C'},\mathbf{W}_{D'}} J_{C',D'}\left(D'(C'(C(\mathbf{O};\mathbf{W}_C);\mathbf{W}_{C'});\mathbf{W}_{D'}),\, K(\mathbf{O})\right); \quad \forall \mathbf{O} \in \mathcal{D} \tag{3.12}$$

, donde la función $K$ se define como:

[5]A cualquier otro pixel se le asigna la clase correspondiente a fondo, $c_7$

$$K(i) = \begin{cases} i & \text{Si } x \in \{c_2, c_3, c_4, c_5\} \\ c_7 & \text{en otro caso} \end{cases} \tag{3.13}$$

, y la función de error $J_{C',D'}$ es la función de entropía cruzada,

$$J_{C',D'} = -\frac{1}{|\mathcal{D}|} \sum_{\mathbf{O} \in \mathcal{D}} \sum_{i=1}^{d} \sum_{j=1}^{n_{c'}} K(\mathbf{O})_{i,j} \ln(\mathbf{C}'_{i,j}) \tag{3.14}$$

, siendo $\mathbf{C} = D'(C'(\vec{z}; \mathbf{W}_{C'}); \mathbf{W}_{D'})$.

El algoritmo 3 describe el proceso de entrenamiento del codificador auxiliar.

**Algoritmo 3** Entrenamiento del codificador auxiliar

**procedure** ENTRENAR_CODIFICADOR_AUXILIAR($\mathbf{W}_C$: parámetros del sensor, $\mathcal{D}$: conjunto de observaciones y matrices de clases, $n_{iters}$: número de iteraciones del entrenamiento, $b$: tamaño de los lotes de entrenamiento, $\alpha_{C'}$: tasa de aprendizaje del codificador auxiliar)
  Inicializar $\mathbf{W}_{C'}$ y $\mathbf{W}_{D'}$
  **for** $n_{iters}$ **do**
    **for** FLOOR($|\mathcal{D}| \div b$) **do**
      Muestrear $b$ elementos uniformemente de $\mathcal{D}$ para obtener lote $\mathcal{B}$
      Calcular:

$$J_{C',D'} = -\frac{1}{b} \sum_{\mathbf{O} \in \mathcal{B}} \sum_{i=1}^{d} \sum_{j=1}^{n_{c'}} K(\mathbf{O})_{i,j} \ln(\mathbf{C}'_{i,j})$$

      Actualizar parámetros del autoencoder:

$$\mathbf{W}_{C'} = \mathbf{W}_{C'} - \alpha_{C'} \nabla_{\mathbf{W}_{C'}} J_{C',D'}$$

$$\mathbf{W}_{D'} = \mathbf{W}_{D'} - \alpha_{C'} \nabla_{\mathbf{W}_{D'}} J_{C',D'}$$

    **end for**
  **end for**
  **return** $\mathbf{W}_{C'}$
**end procedure**

### Modelo social modificado

El modelo social modificado es una función $\phi : \mathbb{R}^{d_z + d_M} \times \mathcal{A}_i \to \mathbb{R}^{d_{z'}}$, cuya función es predecir las acciones del resto de agentes en su campo de visión en el instante siguiente,

dada la observación presente codificada, y la acción del agente. Las ecuaciones 3.7 y 3.8, que describen el problema del aprendizaje del modelo social, quedan para el caso de $N > 2$ agentes como:

$$\min_{\mathbf{W}_\phi} J_\phi \left(\phi(\vec{x}_t, a_t; \mathbf{W}_\phi), \vec{z}_{a_{-i},t+2}\right); \quad \forall(\vec{x}_t, a_t, \vec{z}_{a_{-i},t+2}) \in \mathcal{T}' \tag{3.15}$$

$$J_\phi = -\frac{1}{|\mathcal{T}'|} \sum_{\tau' \in \mathcal{T}'} (\vec{z}_{a_{-i},t+2} - \phi(\vec{x}_t, a_t))^2 \tag{3.16}$$

, donde $\tau' = (\vec{x}_t, a_t, \vec{z}_{a_{-i},t+2})$ y $\mathcal{T}' = \{(\vec{x}_0, a_0, \vec{z}_{a_{-i},2}), ..., (\vec{x}_{L-2}, a_{L-2}, \vec{z}_{a_{-i},L})\}$ es una trayectoria de observaciones codificadas, estados y acciones de longitud $L-1$ obtenidas online por el agente. La función de entropía cruzada de la ecuación 3.8 es reemplazada por el error cuadrado medio en la ecuación 3.16 como función de error puesto que el modelo social ya no estima una distribución de probabilidad.

**Algoritmo 4** Entrenamiento del modelo social modificado

**procedure** ENTRENAR_MODELO_SOCIAL_MODIFICADO($n_{iters}$: número de iteraciones del entrenamiento, $\mathcal{T}'$: trayectoria observaciones-acciones conjuntas codificadas, $\alpha_\phi$: tasa de aprendizaje del modelo social)
  **for** $n_{iters}$ **do**
    Calcular $J_\phi$ (Ecuación 3.16)
    Actualizar parámetros del modelo:

$$\mathbf{W}_\phi = \mathbf{W}_\phi - \alpha_\phi \nabla_{\mathbf{W}_\phi} J_\phi$$

  **end for**
  **return** $\mathbf{W}_\phi$
**end procedure**

### Crítico social modificado

El estimador de la ecuación 3.9 es definido ahora como índice de la información mutua entre las acciones del agente en el instante presente $\mathbf{a}_t$, y el código $\vec{\mathbf{z}}_{a_{-i},t+2}$, que será producto de las acciones en el instante siguiente,

$$\max_{\mathbf{W}_F} \Big\{ \ln(2{,}0) - \mathbb{E}_{(\mathbf{a_t}, \vec{\mathbf{z}}_{a_{-i},t+2}) \sim P_{\mathbf{a_t}\vec{\mathbf{z}}_{a_{-i},t+2}|\vec{\mathbf{x}_t}}} [-\zeta(-F(a_t, \vec{z}_{a_{-i},t+2}; \mathbf{W}_F))] - \\ \mathbb{E}_{(\mathbf{a_t}, \vec{\mathbf{z}}_{a_{-i},t+2}) \sim P_{\mathbf{a_t}|\vec{\mathbf{x}_t}} P_{\vec{\mathbf{z}}_{a_{-i},t+2}|\vec{\mathbf{x}_t}}} [\zeta(F(a_t, \vec{z}_{a_{-i},t+2}; \mathbf{W}_F))] \Big\} \tag{3.17}$$

, por lo que el dominio del discriminador ahora crece a razón de $|\Omega| = |\mathcal{A}_i| + d_{z'}$.

Las muestras de la distribución conjunta $P_{\mathbf{a}_t \vec{\mathbf{z}}_{a_{-i},t+2}|\vec{\mathbf{x}_t}}$ y de la distribución marginal $P_{\mathbf{a_t}|\vec{\mathbf{x}_t}}$ son obtenidas del conjunto de tuplas $\mathcal{T}'$ y de la política del agente, respectivamente. Las muestras de la distribución marginal $P_{\vec{\mathbf{z}}_{a_{-i},t+2}|\vec{\mathbf{x}_t}}$ se obtienen de forma aproximada promediando la salida del modelo social para cada posible acción:

$$\frac{1}{|\mathcal{A}_i|} \sum_{a \in \mathcal{A}_i} \phi(\vec{\mathbf{x}}_t, a) \tag{3.18}$$

Por lo que la función objetivo a maximizar es:

$$J_F = \ln(2{,}0) + \frac{1}{|\mathcal{T}'|} \sum_{\tau' \in \mathcal{T}'} \zeta(-F(\pi(\vec{x}_t, a_t; \mathbf{W}_\pi), \vec{z}_{a_{-i},t+2}; \mathbf{W}_F)) - ... \\ \frac{1}{|\mathcal{T}'|} \sum_{\tau' \in \mathcal{T}'} \zeta \left( F \left( \pi(\vec{x}_t, a_t; \mathbf{W}_\pi), \frac{1}{|\mathcal{A}_i|} \sum_{a \in \mathcal{A}_i} \phi(\vec{x}_t, a); \mathbf{W}_F \right) \right) \tag{3.19}$$

Al igual que con el estimador sin modificar, el crítico social modificado es entrenado junto con la política, como se describe en la siguiente sección.

### 3.1.5. Controlador

El controlador se compone de dos redes neuronales, $\hat{V}(\vec{\mathbf{x}}; \mathbf{W}_V)$ y $\pi(\mathbf{a}, \vec{\mathbf{x}}; \mathbf{W}_\pi)$, que aproximan la política y función de valor del agente, respectivamente, tal como se describió en la sección 2.2.3. Ambas redes neuronales son entrenadas usando un algoritmo basado en el gradiente de la política denominado *Optimización de Política Proximal* [50], que se describe a continuación.

### Optimización de Política Proximal

El algoritmo de Optimización de Política Proximal (PPO[6]) es una modificación al algoritmo básico del gradiente de la política descrito en la sección 2.2.3, que mejora la eficiencia muestral[7] del algoritmo al permitir usar un mismo conjunto de datos durante múltiples iteraciones de entrenamiento, lo que en el algoritmo básico suele conllevar a actualizaciones de gran magnitud de los parámetros y a la no convergencia del entrenamiento [50]. PPO logra esto forzando a que, en cada iteración, la política no diverja drásticamente de una política de referencia.

En PPO la función objetivo de la ecuación 2.13 es reemplazada por la ecuación 3.20,

$$J_\pi = \frac{1}{|\mathcal{E}|} \sum_{(\vec{x},a)\in\mathcal{E}} \text{mín}\left(\rho(\vec{x},a)\hat{A}(\vec{x},a),\ clip\left(\rho(\vec{x},a), 1-\epsilon, 1+\epsilon\right)\hat{A}(\vec{x},a)\right) \tag{3.20}$$

, donde $\mathcal{E} = \{(\vec{x}_0, a_0, r_1), ..., (\vec{x}_{L-1}, a_{L-1}, r_L)\}$ es una trayectoria de experiencias de longitud $L$ obtenidas online por el agente, $\rho(\vec{x},a) = \frac{\pi(a,\vec{x};\mathbf{W}_\pi)}{\pi(a,\vec{x};\mathbf{W}_\beta)}$, $\mathbf{W}_\beta$ son los parámetros de la política de referencia, $\epsilon \in (0,1)$ es un híperparámetro, y la función *clip* se define como:

$$clip(i,a,b) = \begin{cases} a, & \text{si } i < a, \\ i, & \text{si } a \leq i \leq b \\ b, & \text{si } i > b \end{cases} \tag{3.21}$$

, con $a \leq b$.

La función *clip* en la ecuación 3.20 elimina el incentivo para modificar drásticamente la política al hacer nulo el gradiente cuando $\rho(\vec{\mathbf{x}}, \mathbf{a})$ sale del intervalo $[1-\epsilon, 1+\epsilon]$. Se toma el mínimo entre el objetivo acotado y sin acotar de modo que el objetivo final es una cota inferior del objetivo sin acotar.

[6]Proximal Policy Optimization

[7]Cantidad inversamente proporcional al número de datos requeridos para la convergencia.

El estimado de la función de ventaja, $\hat{A}$, se obtiene como se describió en la sección 2.2.3, usando un estimado de la función de valor, $\hat{V}(\vec{x}_t; \mathbf{W}_V)$, cuyos parámetros se sintonizan para resolver de forma aproximada el problema de la ecuación 3.22:

$$\begin{aligned} &\min_{\mathbf{W}_V} J_V \\ J_V(\mathbf{W}_V) &= \sum_{(\vec{x}_t, a_t, r_{t+1}) \in \mathcal{E}} (G(\vec{x}_t; \pi) - \hat{V}(\vec{x}_t; \mathbf{W}_V))^2 \end{aligned} \tag{3.22}$$

El algoritmo de entrenamiento del estimado de la función de valor se describe en el algoritmo 5.

**Algoritmo 5** Entrenamiento del estimado de la función de valor

**procedure** ENTRENAR_FUNCIÓN_DE_VALOR($n_{iters}$: número de iteraciones del entrenamiento, $\mathcal{E}$: conjunto de experiencias, $\alpha_V$: tasa de aprendizaje del estimado de la función de valor)
  **for** $n_{iters}$ **do**
    Calcular $J_V$ (Ecuación 3.22) usando $\mathcal{E}$.
    Actualizar parámetros del estimado de la función de valor:

$$\mathbf{W}_V = \mathbf{W}_V - \alpha_V \nabla_{\mathbf{W}_V} J_V$$

  **end for**
  **return** $\mathbf{W}_V$
**end procedure**

Los autores proponen además el incluir un objetivo de maximización de la entropía de la política, $H(\pi)$, para promover la exploración en la búsqueda de la política y evitar la convergencia temprana a mínimos locales:

$$\max_{\mathbf{W}_\pi} \{J_\pi + w_H H(\pi)\} = \max_{\mathbf{W}_\pi} \left\{ J_\pi - w_H \sum_{(\vec{x}, a, r) \in \mathcal{E}} \pi(a, \vec{x}; \mathbf{W}_\pi) \ln(\pi(a, \vec{x}; \mathbf{W}_\pi)) \right\} \tag{3.23}$$

, donde $w_H$ es una constante que controla el nivel de exploración.

A la función objetivo se añade también como término a maximizar el estimado de la información mutua de la ecuación 3.11(o 3.19 para el caso de más de dos agentes), de forma que los parámetros de la política son sintonizados para maximizar

simultáneamente la ganancia y la información mutua entre las acciones de los agentes. El problema de optimización que define el proceso de entrenamiento de la política es entonces:

$$\begin{aligned} &\max_{\mathbf{W}_\pi, \mathbf{W}_F} \; J \\ &J = J_\pi + w_H H(\pi) + w_F J_F \end{aligned} \tag{3.24}$$

, donde $w_F$ es una constante que controla el peso del objetivo de maximización de la información mutua.

El algoritmo 6 describe el proceso de entrenamiento de la política y del crítico social.

**Algoritmo 6** Entrenamiento de la política y del crítico social

**procedure** ENTRENAR_POLÍTICA($n_{iters}$: número de iteraciones del entrenamiento, $\mathcal{E}$: conjunto de experiencias, $\hat{V}$: estimado de la función de valor, $\mathbf{W}_\pi$: parámetros de la política de entrenamiento, $\mathbf{W}_\beta$: parámetros de la política de referencia, $\mathbf{W}_F$: parámetros del discriminador del crítico social, $\mathbf{W}_\phi$: parámetros del modelo social, $w_H$: peso de la entropía en $J$, $w_F$: peso del estimado de la información mutua en $J$, $\alpha_\pi$: tasa de aprendizaje de la política, $\alpha_F$: tasa de aprendizaje del discriminador del crítico social)

**for** $n_{iters}$ **do**

Calcular estimados de la función de ventaja usando $\mathcal{E}$ y $\hat{V}$ (Ecuación 2.11):

$$\{\hat{A}(\vec{x}_0, a_0), ..., \hat{A}(\vec{x}_{L-1}, a_{L-1})\}$$

Calcular $J$ (Ecuación 3.24)

Actualizar parámetros de la política y del discriminador del crítico social:

$$\mathbf{W}_\pi = \mathbf{W}_\pi + \alpha_\pi \nabla_{\mathbf{W}_\pi} J$$

$$\mathbf{W}_F = \mathbf{W}_F + \alpha_F \nabla_{\mathbf{W}_F} J$$

**end for**

**return** $\mathbf{W}_\pi$, $\mathbf{W}_F$

**end procedure**

## 3.2. Algoritmo de entrenamiento

El algoritmo propuesto para el entrenamiento del sistema multi-agente se presenta en el algoritmo 1. Este se compone a grandes rasgos de dos fases: la fase de inicialización y entrenamiento offline que consiste en la inicialización de todos los parámetros y el entrenamiento del sensor y/o codificador auxiliar, componentes sensoriales comunes a todos los agentes, y la fase de aprendizaje online, en la que los agentes interactúan con el entorno y entre sí, almacenan experiencias, y ajustan sus parámetros para maximizar sus ganancias a largo plazo y su coordinación con otros agentes. A continuación se describe en detalle cada fase.

### 3.2.1. Inicialización y entrenamiento offline

#### Inicialización de los parámetros para cada agente

En este paso cada agente inicializa los parámetros de su modelo social ($\mathbf{W}_\phi$), discriminador del crítico social ($\mathbf{W}_F$), estimado de la función de valor ($\mathbf{W}_V$) y política ($\mathbf{W}_\pi$). La inicialización para cada componente consiste en muestrear cada uno de sus parámetros de una distribución Gaussiana estándar. Los parámetros de la política de referencia ($\mathbf{W}_\beta$) se inicializan con los mismos valores de los de la política principal ($\mathbf{W}_\pi$).

#### Entrenamiento de los codificadores

El sensor y el codificador auxiliar, para el caso de más de dos agentes, son entrenados offline usando un conjunto de observaciones, $\mathcal{D}$, muestreadas uniformemente del espacio de todas las observaciones, $\mathcal{O}$. Para obtener tal conjunto de datos se emplea un sistema de $N$ agentes cuyas políticas son distribuciones uniformes independientes del estado: $\pi(a) = p(\mathbf{a} = a) = \frac{1}{|\mathcal{A}_i|}$; $\forall a \in \mathcal{A}_i$, y se ejecuta en el entorno hasta obtener la cantidad de observaciones deseada. El sensor se entrena según el algoritmo 1, y en el caso de que la población contenga más de dos agentes, o que la cantidad de agentes sea variable en el tiempo, se entrena el codificador auxiliar usando el algoritmo 3.

### 3.2.2. Entrenamiento online

#### Obtención de experiencias

Durante el entrenamiento online cada agente interactúa en paralelo con el entorno, y con el resto de agentes, durante máximo $t_{max} \in [1, \infty)$ instantes de tiempo. El proceso de aprendizaje del agente se da cada $L \ll t_{max}$ instantes de tiempo, durante los cuales el agente almacena trayectorias de experiencias de longitud $L$, siguiendo el proceso descrito por el algoritmo 7 y que se detalla a continuación.

**Algoritmo 7** Obtención de experiencias

**procedure** OBTENER_EXPERIENCIAS($L$: horizonte, $\mathbf{W}_C$: parámetros del sensor, $\mathbf{W}_\beta$: parámetros de la política)
  Inicializar $\vec{h}_t$ de la memoria $M$
  **for** $t = 0, ..., L$ **do**
    **if** $t > 0$ **then**
      Observar acción de otros en su campo de visión $a_{-i,t}$
      Observar recompensa $r_t$
      Guardar tupla $(\vec{x}_{t-1}, a_{t-1}, a_{-i,t}, r_t)$ en $\mathcal{E}$
    **end if**
    Obtener observación $\mathbf{O}_t$ del entorno
    Codificar la observación: $\vec{z}_t = C(\mathbf{O}_t)$
    Computar el estado de la memoria: $\vec{x}_t = [\vec{z}_t, M(\vec{z}_t, \vec{h}_t)]$
    Muestrear acción $a_t$ de $\pi(\mathbf{a}_t, \vec{x}_t; \mathbf{W}_\beta)$ y ejecutarla
  **end for**
  **return** $\mathcal{E}$
**end procedure**

En cada instante de tiempo, el agente recibe una observación, $\mathbf{O}_t$, que es transformada por el sensor a un código representativo, $\vec{\mathbf{z}}_t$ y usada para alterar el estado de la memoria, obteniéndose el vector $\vec{\mathbf{x}}_t$. Acto seguido muestrea una acción de la distribución de probabilidad definida por su política actual, $\mathbf{a}_t \sim \pi(\mathbf{a}_t, \vec{\mathbf{x}}_t; \mathbf{W}_\beta)$, y la ejecuta en el entorno. En respuesta a su acción, y a la acción conjunta del resto de agentes, el agente recibirá en el instante siguiente una recompensa $\mathbf{r}_{t+1}$ y podrá observar la acción que el resto de agentes en su campo de visión ejecutaron, $\mathbf{a}_{-i,t+1}$. En este momento el agente habrá reunido una tupla de experiencia $\tau_t = (\vec{\mathbf{x}}_t, \mathbf{a}_t, \mathbf{a}_{-i,t+1}, \mathbf{a}_{t+1})$ que almacenará en su conjunto de experiencias, $\mathcal{E}$. Tras $L$ instantes de tiempo el agente

ejecuta su rutina de aprendizaje.

**Algoritmo 1** Entrenamiento del sistema multi-agente

**for each** agente **do**
    Inicializar parámetros de las redes neuronales
**end for**
Obtener conjunto de datos para el sensor
Entrenar sensor
$t_{total} = 0$
**while** $t_{total} \leq t_{max}$ **do**
    **for each** agente **do in parallel**
        Interactuar con el entorno por $L$ instantes para obtener experiencias.
        Entrenar modelo social $\phi$
        Entrenar crítico social $\hat{I}$
        Entrenar función de valor $\hat{V}$
        Usar estimado $\hat{V}$ y $\hat{I}$ para entrenar la política $\pi$
        $t_{total} = t_{total} + L$
    **end for**
**end while**

### Rutina de aprendizaje del agente

Una vez el agente ha adquirido suficientes experiencias con su política actual procede a procesarlas para sintonizar sus parámetros y obtener una nueva política. Este proceso consiste de varios pasos:

1. **Entrenamiento del modelo social**: el modelo social, estándar para el caso de dos agentes o modificado para el caso de más de dos agentes o de poblaciones de composición variante en el tiempo, es entrenado usando el algoritmo 2 o 4 según sea el caso. Las tuplas de observación codificadas y estados, $\vec{\mathbf{x}}_t$, y acciones de otros agentes, $\mathbf{a}_{-i,t+1}$ o $\vec{\mathbf{z}}_{a_{-i},t+2}$, son obtenidas del conjunto de experiencias $\mathcal{E}$.

2. **Entrenamiento del estimado de la función de valor**: el estimado de la función de valor es entrenado según el algoritmo 5. Las tuplas de observaciones codificadas y estados, $\vec{\mathbf{x}}_t$, y recompensas obtenidas, $\mathbf{r}_{t+1}$, son obtenidas del conjunto de experiencias $\mathcal{E}$.

3. **Entrenamiento de la política**: la política es actualizada según el algoritmo 6. Este paso involucra todos los componentes del agente. El estimado de la función de valor, $\hat{V}$, es usado junto con la trayectoria de recompensas en $\mathcal{E}$ para calcular un estimado de la función de ventaja que actúa como crítico en la actualización de la política para guiarla hacia mayores ganancias. La política usada durante la obtención de experiencias es usada como política de referencia para evitar modificaciones drásticas a la política principal. El crítico social es usado para estimar la información mutua entre las acciones del agente y las acciones de sus pares haciendo uso de $F$, el discriminador entre muestras provenientes de la distribución conjunta contenidas en $\mathcal{E}$ y muestras del producto las distribuciones marginales, obtenidas marginalizando el efecto de la acción del agente de la predicción hecha por el modelo social, $\phi$. El crítico social guía la actualización de la política a acciones más coordinadas con el resto de agentes. En este paso los parámetros del discriminador $F$ son también actualizados de modo que se mejore la precisión del crítico social, y/o que se adapte a los cambios en el tiempo de las distribuciones de las acciones de los agentes.

Finalmente, los parámetros de la política de interacción, $\mathbf{W}_\beta$, se hacen iguales a los obtenidos tras la actualizacion de la política principal, $\mathbf{W}_\pi$, con lo que se obtiene una política actualizada que será usada en la obtención de nuevas experiencias en el siguiente episodio.

## 3.3. Índices de desempeño del sistema

A diferencia de los problemas de aprendizaje por refuerzo de un solo agente donde la función de valor es un índice suficiente de desempeño, en el caso de un problema multi-agente con objetivos divergentes no representa adecuadamente el comportamiento grupal [40]. En este trabajo se usan como índices macroscópicos los *índices sociales* de *utilidad*, *equidad*, *paz*, y *sostenibilidad* propuestos en [40] para el juego de la cosecha, y se define un índice de *cooperación* específico al problema basado en la definición dada en la sección 2.1.

En un sistema de $N$ agentes en el juego de la cosecha, sean $\{r_{i,t} \,|\, t = 1, ..., L\}$ y $\{\mathbf{O}_{i,t} \,|\, t = 1, ..., L\}$ la secuencia de recompensas y observaciones respectivamente, del $i$-ésimo agente a lo largo de un episodio de longitud $T$, y sea $G_i$ su ganancia no descontada,

$$G_i = \sum_{t=0}^{L-1} r_{i,t} \tag{3.25}$$

, se definen entonces los siguientes índices macroscópicos:

**Utilidad**

Es el promedio sobre agentes de la ganancia obtenida:

$$U = \frac{1}{N} \sum_{i=1}^{N} G_i \tag{3.26}$$

**Equidad**

Mide la dispersión en la distribución de la ganancia entre agentes. Se define como:

$$E = 1 - g \tag{3.27}$$

, donde $g$ es el *coeficiente de Gini*, un índice de desigualdad [51]:

$$g = \frac{\sum_{i=1}^{N} \sum_{j=1}^{N} |G_i - G_j|}{2N \sum_{i=1}^{N} G_i} \tag{3.28}$$

**Paz**

Se define como el tiempo promedio sin agentes fuera de juego:

$$P = 1 - \frac{\sum_{i=1}^{N} \sum_{t=1}^{L} \mathbb{1}(\mathbf{O}_{i,t} \,;\, \mathbf{O}_{i,t} = \mathbf{O}_{TO})}{NL} \tag{3.29}$$

, donde $\mathbf{O}_{TO}$ es la observación que recibe un agente cuando es impactado por el rayo de tiempo fuera.

**Sostenibilidad**

Es la cantidad de manzanas en el campo de juego a lo largo del episodio:

$$S = \sum_{t=1}^{L} \sum_{i} \mathbb{1}(\mathbf{S}_{t,i}\,;\, \mathbf{S}_{t,i} = \text{manzana}) \tag{3.30}$$

, donde $\mathbf{S}_{t,i}$ es el $i$-ésimo píxel del estado en el instante $t$.

**Cooperación**

El índice de cooperación según como se definió en la sección 2.1 se define para el juego de la cosecha como:

$$c = \frac{\bar{I}}{\bar{H}}\, U \tag{3.31}$$

, donde la utilidad, $U$ es usada como índice de desempeño, y el índice de correlación es la razón entre la información mutua promedio estimada,

$$\bar{I} = \frac{1}{N} \sum_{i=1}^{N} \hat{I}_i \tag{3.32}$$

, y la entropía promedio en el sistema,

$$\bar{H} = \frac{1}{N} \sum_{i=1}^{N} H(\pi_i) \tag{3.33}$$

, siendo $\pi_i$ la política del $i$-ésimo agente, y $J_{F_i}$ el estimado de la información mutua (ecuación 3.11 o ecuación 3.19) del $i$-ésimo agente. La normalización por la entropía promedio se hace para mitigar la dependencia del índice de correlación de la cantidad de incertidumbre en el sistema.

# Capítulo 4

# Experimentos y resultados

En este capítulo se describen los experimentos realizados, la metodología de experimentación, y se presentan y analizan los resultados obtenidos del entrenamiento del sistema multi-agente. El entrenamiento se hace para dos diferentes sistemas, uno de dos agentes y uno de diez agentes, en dos diferentes entornos. Los dos sistemas son entrenados con el algoritmo que incluye el objetivo de maximización de la información mutua, y un algoritmo sin este objetivo, que se usa como referencia para evaluar el sistema propuesto en el capítulo 3. Los resultados se presentan como la evolución temporal de los sistemas según los índices definidos en la sección 3.3 y la caracterización del valor final de estos índices tras el entrenamiento en el conjunto de experimentos.

## 4.1. Entrenamiento offline

En esta etapa de la experimentación se realiza la obtención de datos y el entrenamiento offline para obtener el sensor y el codificador auxiliar, componentes comunes a todos los agentes del sistema multi-agente. Primero se describe el experimento por el cual se obtuvieron los datos para el entrenamiento de los dos autoencoders, luego, para cada componente, se describen los parámetros del entrenamiento y se muestra la evolución temporal de la precisión de reconstrucción de su decodificador en el conjunto de datos de entrenamiento y validación.

### 4.1.1. Obtención de datos

La obtención de datos consiste como se describió en la sección 3.2 de la ejecución de un sistema multi-agente que ejecuta acciones en el entorno según una distribución uniforme. El entorno escogido fue el juego de la cosecha en el *mapa abierto grande* descrito en [40] y que se ilustra en la figura 4-1. El sistema multi-agente para la recolección de datos consistió de 10 agentes, y fue ejecutado en el entorno durante 1280 episodios de 1000 instantes de tiempo, de modo que se obtuvieron $1{,}28 \times 10^7$ datos para el entrenamiento de los codificadores. Los datos son particionados en un conjunto de entrenamiento de $1 \times 10^7$ datos y un conjunto de validación de $2{,}8 \times 10^6$ datos seleccionados aleatoriamente con probabilidad uniforme.

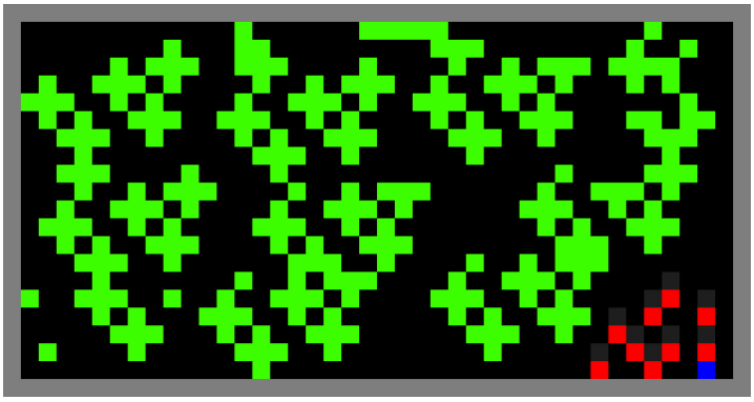

Figura 4-1: Mapa abierto grande.

### 4.1.2. Entrenamiento del sensor

El sensor fue implementado como una red neuronal compuesta de capas convolucionales[1] y capas densamente conectadas. La arquitectura del codificador ($C$) y el decodificador ($D$) del sensor se describe en la tabla 4.1. El sensor es entrenado usando los datos obtenidos por el proceso descrito en la sección 4.1.1 mediante el algoritmo de gradiente de descenso estocástico por lotes con el optimizador *Adam* [52]. Los parámetros del algoritmo de entrenamiento se muestran en la tabla 4.2.

[1]Para información de redes neuronales convolucionales consultar la referencia [5]

| | Capa | Dimensión | Función de activación |
|---|---|---|---|
| $C$ | 1. Entrada | $9 \times 9 \times 7$ | · |
| | 2. Capa convolucional | $(3 \times 3, 16)$ | ReLU |
| | 3. Capa de redimensionamiento | 784 | · |
| | 4. Capa densamente conectada | 32 | ReLU |
| | 5. Capa densamente conectada | 32 | Lineal |
| $D$ | 6. Capa densamente conectada | 32 | ReLU |
| | 7. Capa densamente conectada | 784 | ReLU |
| | 8. Capa de redimensionamiento | $7 \times 7 \times 16$ | · |
| | 9. Capa convolucional transpuesta | $(3 \times 3, 7)$ | SoftMax |

Tabla 4.1: Arquitectura de la red neuronal que implementa el sensor. La tupla en la dimensión de una capa convolucional corresponde a: (tamaño del filtro, número de filtros).

| | Parámetro | Valor |
|---|---|---|
| | Número de experimentos independientes | 1 |
| | Tamaño de los lotes ($b$) | 128 |
| $C$ | Número de epochs ($n_{iters_C}$) | 10 |
| | Tasa de aprendizaje ($\alpha_C$) | $1 \times 10^{-3}$ |
| $C'$ | Número de epochs ($n_{iters_{C'}}$) | 10 |
| | Tasa de aprendizaje ($\alpha_{C'}$) | $1 \times 10^{-3}$ |

Tabla 4.2: Parámetros del algoritmo de entrenamiento del sensor y del codificador auxiliar.

### Resultados

Durante el entrenamiento del sensor se midió la precisión en la reconstrucción del decodificador en los conjuntos de datos de entrenamiento y de validación. La figura 4-2 muestra la curva de error porcentual en el conjunto de entrenamiento en cada lote (izquierda) y en el conjunto de validación en cada epoch (derecha) a lo largo del entrenamiento. El error porcentual es calculado como:

$$\text{Error porcentual} = \frac{\#\text{ de muestras reconstruidas con errores}}{\text{Tamaño del conjunto de muestras}} \times 100\,\% \tag{4.1}$$

Se observa que tanto para el conjunto de entrenamiento como para el conjunto

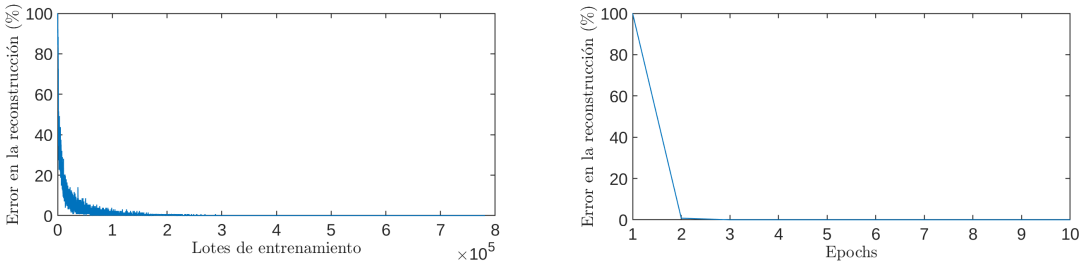


Figura 4-2: Curvas de error porcentual de reconstrucción para el decodificador del sensor a lo largo del entrenamiento en el conjunto de entrenamiento (izquierda) y de validación (derecha).

de validación el error de reconstrucción se vuelve nulo tras el cuarto epoch, por lo que se considera que el sensor ha aprendido a representar adecuadamente cualquier observación del entorno, y por tanto no fue necesario realizar más de un experimento para su entrenamiento.

### 4.1.3. Entrenamiento del codificador auxiliar

El codificador auxiliar es una red neuronal compuesta de capas densamente conectadas, y su decodificador hace uso de capas convolucionales y densamente conectadas. La arquitectura del codificador ($C'$) y del decodificador ($D'$) del codificador auxiliar se describe en la tabla 4.3. El codificador auxiliar es entrenado usando el mismo algoritmo de optimización que el sensor con los parámetros de la tabla 4.2.

| | Capa | Dimensión | Función de activación |
|---|---|---|---|
| $C'$ | 1. Entrada | 32 | · |
| | 2. Capa densamente conectada | 256 | ReLU |
| | 3. Capa densamente conectada | 128 | ReLU |
| | 4. Capa densamente conectada | 16 | Lineal |
| $D'$ | 5. Capa densamente conectada | 128 | ReLU |
| | 6. Capa densamente conectada | 784 | ReLU |
| | 7. Capa de redimensionamiento | $7 \times 7 \times 16$ | · |
| | 8. Capa convolucional transpuesta | $(3 \times 3, 5)$ | SoftMax |

Tabla 4.3: Arquitectura de la red neuronal que implementa el codificador auxiliar. La tupla en la dimensión de una capa convolucional corresponde a: (tamaño del filtro, número de filtros).

### Resultados

La figura 4-3 muestra la curva de error porcentual de reconstrucción para el decodificador del codificador auxiliar en el conjunto de entrenamiento en cada lote (izquierda) y en el conjunto de validación en cada epoch (derecha). El error de reconstrucción se hace nulo tras el sexto epoch en ambos conjuntos de datos. En este punto se considera que el codificador ha aprendido a representar adecuadamente cualquier observación del entorno, siendo innecesario realizar más de un experimento para su entrenamiento.

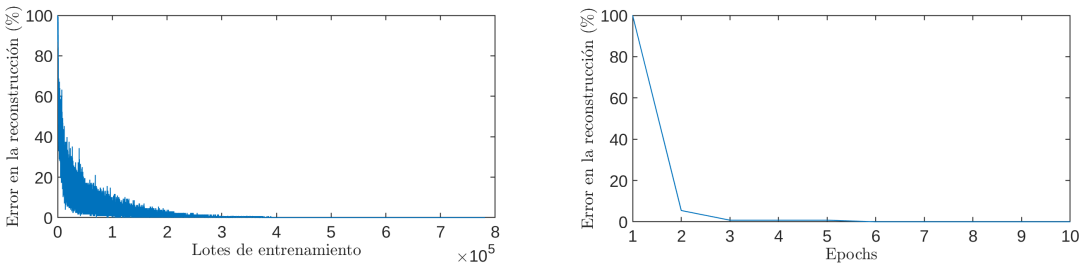


Figura 4-3: Curvas de error porcentual de reconstrucción para el decodificador del codificador auxiliar a lo largo del entrenamiento en el conjunto de entrenamiento (izquierda) y de validación (derecha).

## 4.2. Entrenamiento online

En esta etapa de la experimentación se entrena el sistema multi-agente en el entorno, se mide su desempeño según los índices descritos en la sección 3.3, y se caracteriza la estrategia del sistema según los índices de desempeño y la dinámica de la información en el sistema. Dos sistemas multi-agente de diferentes tamaños son entrenados en dos diferentes entornos. Primero, se entrena un sistema de 2 agentes en el *mapa pequeño* del juego de la cosecha (Figura 2-2). Luego, se entrena un sistema de 10 agentes en el *mapa abierto grande* del juego de la cosecha (Figura 4-1). El sistema de 2 agentes en el mapa pequeño y de 10 agentes en el mapa abierto grande se escogieron dado que en [39] y [40], respectivamente, se demostró empíricamente que estos sistemas en particular cumplían con las características de un dilema social secuencial.

La arquitectura de las redes neuronales de los agentes en ambos casos es la descrita

por la tabla 4.4, donde el radio espectral de la matriz de pesos de la capa recurrente de la memoria es de $\rho_1 = 0{,}95$, y es inicializada al mismo valor para todos los agentes. El sensor y codificador auxiliar son comunes a todos los agentes y son los obtenidos del entrenamiento offline de la sección 4.1.

Para ambas configuraciones del entorno y del tamaño de la población de agentes se realizan 30 experimentos independientes en los que se entrena un *sistema de referencia* entrenado con la función objetivo estándar del algoritmo PPO (ecuación 3.23), y un sistema multi-agente con el objetivo de maximización de la información mutua entre agentes (ecuación 3.24). De ahora en adelante, por brevedad, a este último sistema se le llamará *sistema acoplado*, debido al acople explícito que existe entre las dinámicas de las políticas de los agentes. Cada sistema multiagente es entrenado durante 10 millones de instantes de tiempo en episodios de máximo 1000 instantes. Las mediciones de los índices de desempeño se toman cada episodio. La tabla 4.5 muestra los parámetros de experimentación y los parámetros de entrenamiento, que fueron escogidos según valores previamente usados en la literatura [42, 50, 53].

| | Capa | Dimensión | Función de activación |
|---|---|---|---|
| Memoria ($M$) | 1. Entrada | 32 | · |
| | 2. Capa recurrente | 1024 | tanh |
| Modelo social ($\phi$) | 1. Entrada | 1056 | · |
| | 2. Capa densamente conectada | 8 (ó 16) | SoftMax (ó lineal) |
| Discriminador DV ($F$) | 1. Entrada | 16 | · |
| | 2. Capa densamente conectada | 32 | ReLU |
| | 3. Capa densamente conectada | 32 | ReLU |
| | 4. Capa densamente conectada | 1 | Lineal |
| Función de valor ($\hat{V}$) | 1. Entrada | 1056 | · |
| | 2. Capa densamente conectada | 1 | Lineal |
| Política ($\pi$) | 1. Entrada | 1056 | · |
| | 3. Capa densamente conectada | 8 | SoftMax |

Tabla 4.4: Arquitectura de las redes neuronales del agente.

| | Parámetro | Valor |
|---|---|---|
| Experimentación | Número de experimentos independientes | 30 |
| | Duración del experimento ($t_{max}$) | $1 \times 10^7$ |
| | Duración máxima de un episodio ($L$) | 1000 |
| | Número de epochs ($n_{iters}$) | 5 |
| Modelo social ($\phi$) | Tasa de aprendizaje ($\alpha_\phi$) | $1 \times 10^{-3}$ |
| Función de valor ($\hat{V}$) | Tasa de aprendizaje ($\alpha_V$) | $1 \times 10^{-3}$ |
| Discriminador DV ($F$) | Tasa de aprendizaje ($\alpha_F$) | 0.01 |
| Política ($\pi$) | Tasa de aprendizaje ($\alpha_\pi$) | $1 \times 10^{-4}$ |
| | Peso de penalización de IM ($w_F$) | 0.1 |
| | Peso de penalización de entropía ($w_H$) | 0.01 |

Tabla 4.5: Parámetros de experimentación y entrenamiento.

### 4.2.1. Resultados en el sistema de dos agentes

#### Evolución temporal del sistema

La figura 4-4 muestra la evolución temporal de los índices de utilidad, equidad, paz y sostenibilidad para los 30 experimentos del sistema de dos agentes. Es evidente de la gráfica que el sistema acoplado supera al sistema de referencia en los cuatro índices durante la mayor parte del entrenamiento.

Durante los primeros 2 millones de instantes, los índices sociales para ambos sistemas exhiben una dinámica similar a la descrita en [40]. Inicialmente los agentes pasan por una fase de aprendizaje durante la cual aprenden a aumentar la rapidez con la que cosechan los recursos a su alcance. Esta fase se caracteriza por el crecimiento del índice de utilidad y el descenso continuo del índice de sostenibilidad. La equidad crece a medida que los agentes aprenden políticas que les brindan ganancias similares. Estos aprenden además a no usar el rayo de tiempo fuera puesto que no acarrea recompensas inmediatas, lo que se manifiesta en el crecimiento del índice de paz. En los cuatro índices el sistema acoplado presenta una dinámica más lenta que el sistema de referencia. Esto podría deberse a que el objetivo de maximización de la información mutua también incentiva la maximización de la entropía, ralentizando la convergencia.

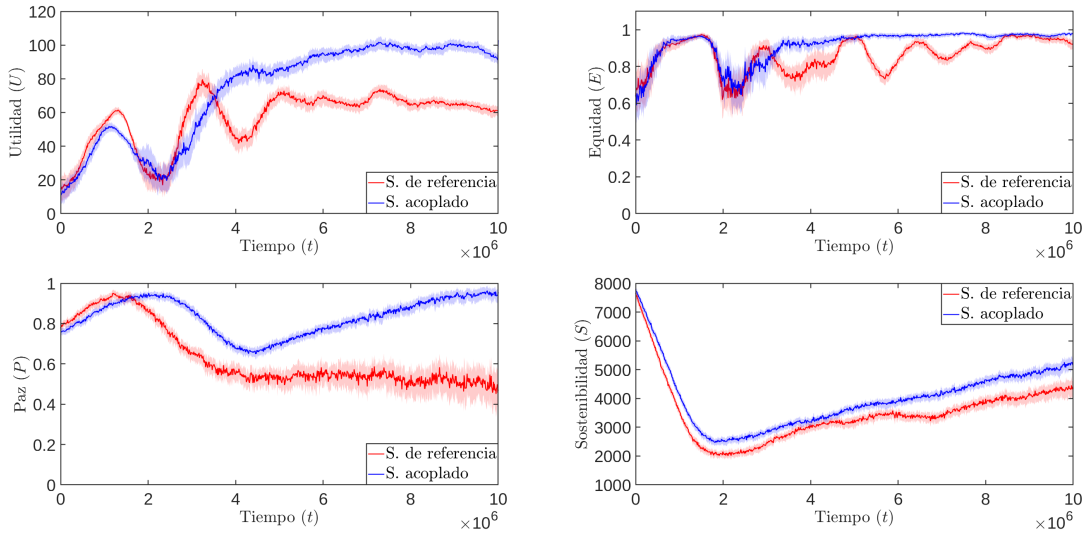


Figura 4-4: Evolución temporal de los índices sociales en el sistema de dos agentes. La línea sólida es la media muestral obtenida de los 30 experimentos, y las barras de error (sombreado) muestran un intervalo de confianza del 99.5 %.

Alrededor de $t = 1{,}32 \times 10^6$ los agentes han aprendido a cosechar a una velocidad mayor a la velocidad con que se regenera el recurso, es decir, el recurso es sobre-explotado. En este punto la utilidad comienza a descender, alcanzando un punto mínimo, junto con la sostenibilidad y la equidad, cerca del instante $t = 2{,}33 \times 10^6$. Una vez los recursos empiezan a escasear los agentes descubren que usando el rayo de tiempo fuera pueden aumentar la cantidad de recursos a su disposición sacando a otros agentes del juego. Esto se manifiesta en el descenso acelerado del índice de paz después de los primeros dos millones de instantes.

Tras $t = 2{,}42 \times 10^6$ la utilidad y sostenibilidad empiezan nuevamente a crecer, sin embargo los índices de equidad y paz evidencian que las causas de esta dinámica a la alza no son las mismas para ambos sistemas. El índice de paz tras los primeros dos millones de instantes decrece para el sistema de referencia con una pendiente aproximada de $-0{,}1493 \times 10^{-6}$ por cada instante de tiempo hasta aproximadamente $t = 4{,}29 \times 10^6$, punto a partir del cual el descenso continúa, mucho más lentamente, con una pendiente aproximada de $-8{,}94 \times 10^{-9}$. El descenso de la paz indica que el mecanismo por el cual el sistema reduce la sobre-explotación del recurso es la reduc-

ción de la población efectiva mediante el uso del rayo de tiempo fuera. La presencia del conflicto entre agentes también se manifiesta en el índice de equidad, que presenta oscilaciones semi-periódicas, indicando que alguno de los dos agentes durante ciertos periodos del entrenamiento es explotado por el otro, posiblemente debido a que alguno de los dos desarrolla una estrategia violenta más eficaz. El agente explotado, al experimentar menos instantes de tiempo efectivos, recibe también menos experiencias, lo que disminuye su velocidad de adaptación a la estrategia de su contrincante y posiblemente justifica el largo periodo de las oscilaciones. Los puntos máximos en las oscilaciones de la equidad serían entonces puntos en que ambos agentes se han adaptado a la estrategia del otro, sin embargo la prevalencia de las oscilaciones sugiere que este no es un punto asintóticamente estable.

En el sistema acoplado el índice de paz también decrece inicialmente, pero en $t = 4{,}38 \times 10^6$ empieza a aumentar, tendencia que permanece hasta el final del entrenamiento cuando alcanza un punto máximo de $P = 0{,}9556$. El índice de equidad crece y alcanza en $t = 4{,}65 \times 10^6$ un valor promedio de $E = 0{,}97 \pm 2\,\%$ que mantiene hasta el final del experimento. El aumento de la paz y equidad a la par de la sostenibilidad indica que los agentes aprenden a cooperar para cosechar sin sobre-explotar el recurso, lo que, en concordancia con la primera desigualdad de los dilemas sociales (sección 2.4), resulta en una utilidad mayor al final del entrenamiento, $U = 102{,}9708$, que la que obtiene el sistema multi-agente de referencia con la estrategia no cooperativa, $U = 61{,}2673$.

La dinámica de la información a lo largo del experimento también brinda perspectivas valiosas para entender la dinámica del sistema. La figura 4-5 muestra la evolución temporal de la entropía promedio y del estimado de la información mutua promedio en la población.

La entropía promedio para ambos sistemas exhibe una dinámica similar, inicialmente descendiendo a medida que las políticas convergen a la estrategia de sobre-explotación, luego creciendo al adaptarse a la escasez de recursos, y finalmente descendiendo nuevamente a medida que las políticas convergen a estrategias cooperativas y competitivas para el sistema acoplado y de referencia, respectivamente. El descenso

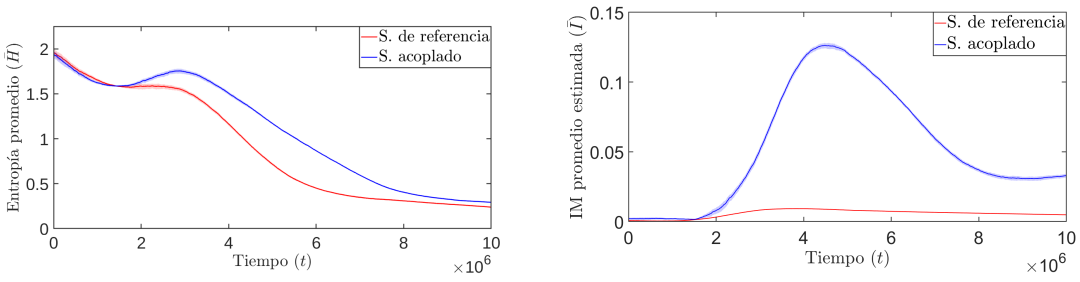


Figura 4-5: Evolución temporal de la información en el sistema de dos agentes. La línea sólida es la media muestral obtenida de los 30 experimentos, y las barras de error (sombreado) muestran un intervalo de confianza del 99.5 %.

de la entropía en el sistema acoplado es más lento que en el sistema de referencia debido a que el objetivo de maximización de la información mutua incentiva a los agentes a maximizar también la entropía.

En ambos sistemas la información mutua inicialmente es casi nula, es decir, las acciones de los agentes tienden a ser independientes. Tras $t = 2 \times 10^6$ la información mutua presente una dinámica a la alza, lo que podría explicarse si se considera que hasta antes de $t = 2 \times 10^6$ ignorar la presencia de otros agentes y enfocarse en el recurso era una buena estrategia, lo que implicaría que el gradiente de la función de ganancia dirigiría a la política a regiones del espacio de búsqueda con baja información mutua con otros agentes. Sin embargo, una vez la velocidad de consumo supera la capacidad de regeneración del recurso, el entorno se vuelve competitivo, y los agentes requieren considerar a sus pares para competir más eficientemente, lo que produciría el alza de la información mutua en ambos sistemas.

En el sistema acoplado la emergencia de acciones coordinadas que generen ganancias causaría que el gradiente del crítico social y el gradiente de la función de ganancia ahora guiaran la política a regiones similares del espacio de búsqueda, causando el aumento acelerado de la información mutua. Inicialmente esta información parece ser usada para competir por el recurso, según el descenso del índice de paz, sin embargo tras $t = 4{,}38 \times 10^6$, punto en que el estimado promedio de la información mutua alcanza su valor máximo, $\bar{I} = 0{,}1262$, el índice de paz comienza a ascender, indicando la transición de una estrategia competitiva a una estrategia cooperativa.

En la figura 4-6 se muestra la evolución temporal del índice de cooperación. Se observa que el índice de cooperación logra capturar diversos atributos que intuitivamente se darían a un sistema cooperativo en el juego de la cosecha, como el ascenso de la utilidad, el ascenso de la equidad a la par de la sostenibilidad, y la transición del conflicto a la paz a medida que la cooperación alcanza un aparente valor de establecimiento. La dinámica a la alza de la información mutua, y la consecuente emergencia de cooperación, sería el mecanismo por el cual en este punto divergen las dinámicas del sistema acoplado y del sistema de referencia, resultando en el desempeño superior del sistema acoplado.

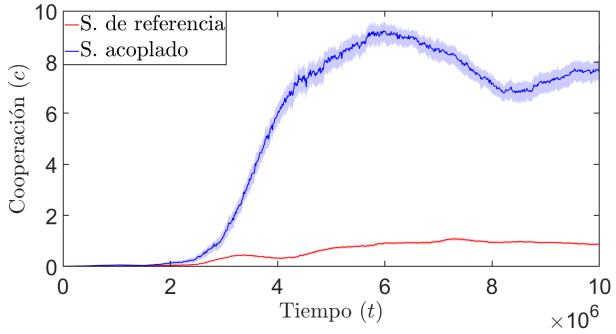


Figura 4-6: Evolución temporal del índice de cooperación en el sistema de dos agentes. La línea sólida es la media muestral obtenida de los 30 experimentos, y las barras de error (sombreado) muestran un intervalo de confianza del 99.5 %.

Tras alcanzar su punto máximo la información mutua promedio empieza a descender obedeciendo al descenso de la entropía, puesto que el mínimo entre las entropías de un conjunto de variables aleatorias es una cota superior para su información mutua. En este caso:

$$\bar{I} \leq \text{mín}(H(\pi_1), ..., H(\pi_N)) \tag{4.2}$$

### Resultados del entrenamiento

El resultado del proceso de entrenamiento del sistema de dos agentes se caracteriza observando la diferencia entre la distribución de los índices de desempeño en el primer y último episodio del entrenamiento para los 30 experimentos. La figura 4-7 muestra

los histogramas de los cuatro índices de desempeño en el primer (columna izquierda) y último (columna derecha) episodio para el sistema acoplado y el sistema de referencia, y la tabla 4.6 compara la media y varianza de los índices de desempeño para los dos sistemas al comienzo y final del entrenamiento.

| | | Sistema de referencia | | Sistema acoplado | |
|---|---|---|---|---|---|
| | Índice | Media | Varianza | Media | Varianza |
| Inicial | Utilidad | 14.6433 | 181.3565 | 11.3877 | 130.3517 |
| | Equidad | 0.6371 | 0.0299 | 0.6457 | 0.0374 |
| | Paz | 0.7766 | 0.0011 | 0.7531 | 0.0007 |
| | Sostenibilidad | $7{,}6647 \times 10^3$ | $3{,}1754 \times 10^4$ | $7{,}7864 \times 10^3$ | $2{,}9929 \times 10^5$ |
| | Cooperación | 0.0028 | $9{,}2055 \times 10^{-6}$ | 0.0093 | 0.0001 |
| Final | Utilidad | 61.2673 | 23.1289 | **102.9708** | **18.3405** |
| | Equidad | 0.9238 | 0.0018 | **0.9814** | **0.0002** |
| | Paz | 0.4720 | 0.0286 | **0.9556** | **0.0026** |
| | Sostenibilidad | $4{,}3532 \times 10^3$ | $2{,}6412 \times 10^5$ | $\mathbf{5{,}1511 \times 10^3}$ | $\mathbf{1{,}4238 \times 10^5}$ |
| | Cooperación | 0.5846 | **0.0024** | **8.6556** | 0.6131 |

Tabla 4.6: Comparación de los índices de desempeño y del índice de cooperación entre el sistema acoplado y el sistema de referencia al inicio y final del entrenamiento en el sistema de dos agentes.

Tanto en los histogramas de la figura 4-7 como en la tabla 4.6 se puede apreciar que en el sistema de referencia la media de la distribución aumenta con el proceso de entrenamiento, salvo en el índice de paz y el índice de sostenibilidad, y la varianza se reduce para los índices de utilidad y equidad. En el sistema acoplado la media aumenta para todos los índices salvo el índice de sostenibilidad, y la varianza se reduce para los índices de utilidad, equidad y sostenibilidad. Considerando la utilidad como la medida de desempeño, y según la definición de aprendizaje dada en la sección 2.1, se puede decir que el sistema multi-agente ha aprendido en ambos casos.

La diferencia de las distribuciones de los índices de desempeño entre el sistema de referencia y el sistema acoplado en el último episodio también indica la clara superioridad del sistema acoplado respecto al sistema de referencia, obteniendo una media superior en todos los índices, como se indica con los valores en negrilla en la tabla 4.6, y una menor varianza en todos los índices salvo el índice de cooperación.

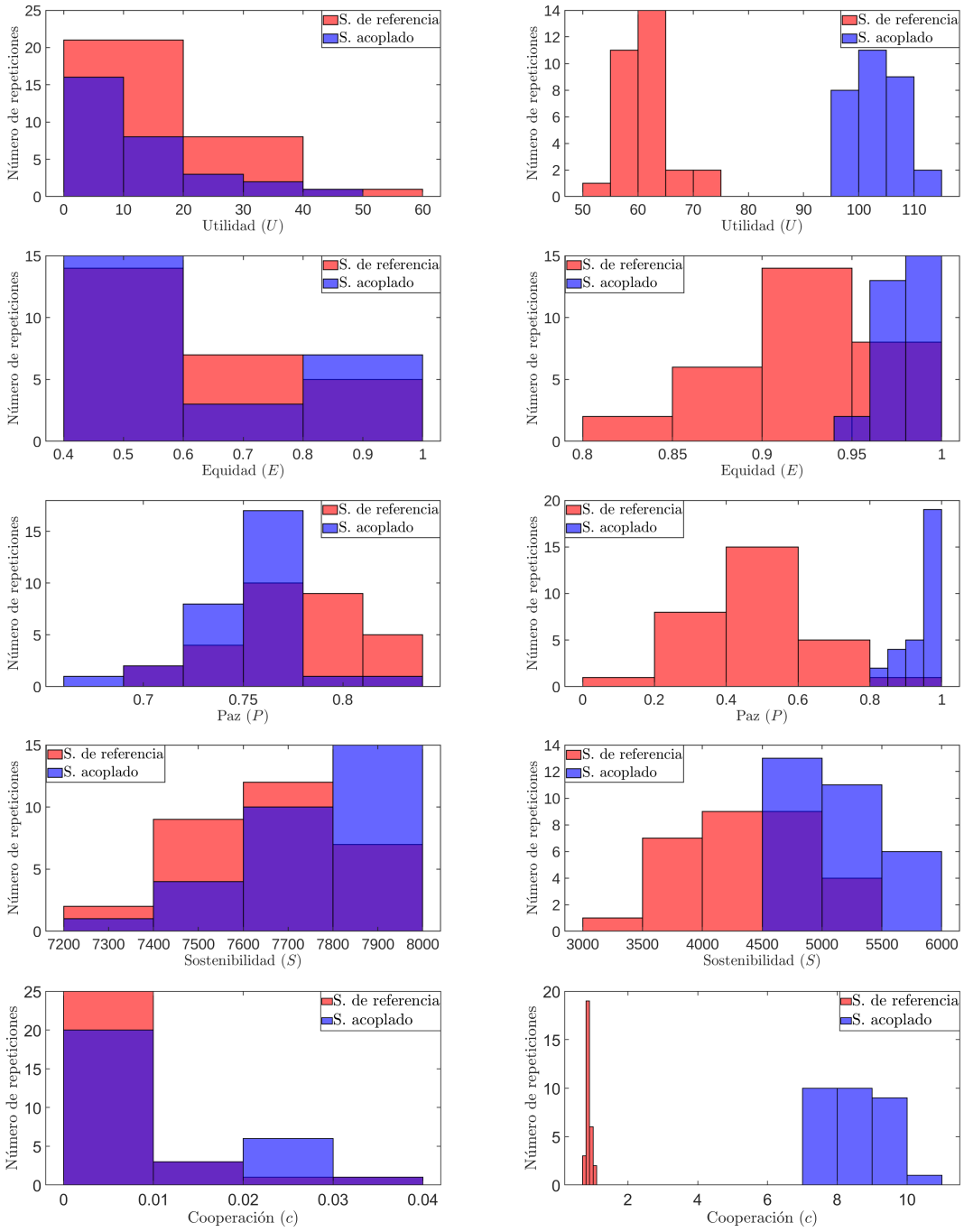


Figura 4-7: Histogramas de los índices de desempeño y del índice de cooperación en el primer episodio (izquierda) y último episodio (derecha) del entrenamiento en el sistema de dos agentes.

### 4.2.2. Resultados en el sistema de diez agentes

#### Evolución temporal

La figura 4-8 muestra la evolución temporal de los índices de utilidad, equidad, paz y sostenibilidad para los 30 experimentos del sistema de diez agentes. La dinámica observada es muy similar a la obtenida en el caso de dos agentes, evidenciándose que con el tiempo el sistema acoplado supera al sistema de referencia en los cuatro índices.

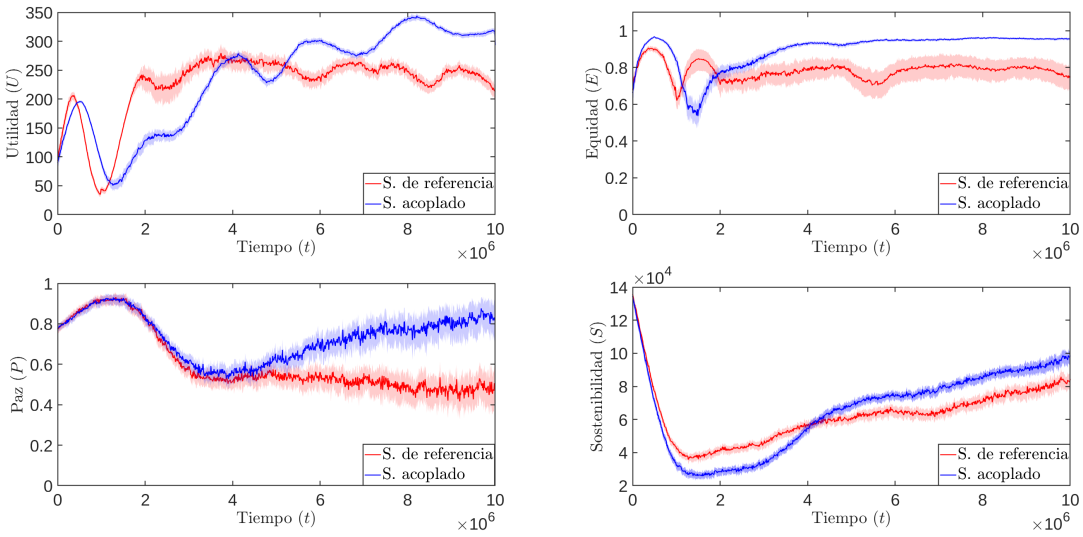


Figura 4-8: Evolución temporal de los índices sociales en el sistema de diez agentes. La línea sólida es la media muestral obtenida de los 30 experimentos, y las barras de error (sombreado) muestran un intervalo de confianza del 99.5 %.

En una primera fase el sistema pasa por la fase de aprendizaje de estrategias de sobre-explotación, en la que la utilidad aumenta y la sostenibilidad disminuye a medida que los agentes aprenden a cosechar con mayor velocidad. Esta tendencia se mantiene hasta $t = 5 \times 10^5$, punto en el que la velocidad de consumo del recurso supera su capacidad de regeneración, lo que genera el descenso de la utilidad.

Tras este punto en ambos sistemas aparecen estrategias competitivas, en las que los agentes usan el rayo de tiempo fuera para reducir la población efectiva y poder explotar el recurso egoístamente, como se observa por el descenso del índice de paz y el aumento del índice de sostenibilidad. En esta etapa el índice de equidad presenta

mayor varianza en el sistema de referencia que en el sistema acoplado, indicando que aún en el conflicto la mayor coordinación entre las acciones de los agentes en el sistema acoplado resulta en recursos repartidos más equitativamente. La equidad en el sistema de referencia no presenta las oscilaciones semi-periódicas observadas en el sistema de dos agentes, esto podría deberse a que dada la mayor cantidad de agentes y el tamaño del entorno es menos probable que haya una relación dominante-dominado que persista. Al igual que en el caso del sistema de dos agentes se observa que la dinámica del sistema acoplado en todos los índices es más lenta.

El sistema de referencia mantiene la estrategia competitiva en el resto del entrenamiento. En el sistema acoplado en cambio, en $t = 3{,}9 \times 10^6$ el índice de paz pasa a una dinámica a la alza, que continúa hasta el final del entrenamiento, alcanzando un valor promedio máximo de $P = 0{,}8157$, e indicando una estrategia cooperativa, según lo cuantifica el índice de cooperación (figura 4-9). La estabilización del índice de equidad en un valor de $E = 0{,}96 \pm 1\,\%$, y el aumento en la velocidad de crecimiento del índice de sostenibilidad en $t = 3{,}06 \times 10^6$ también son indicadores de la transición a una estrategia cooperativa. El uso de una estrategia cooperativa en el sistema acoplado resulta en una utilidad mayor que la obtenida por la estrategia competitiva del sistema de referencia, obedeciendo la primera desigualdad de los dilemas sociales.

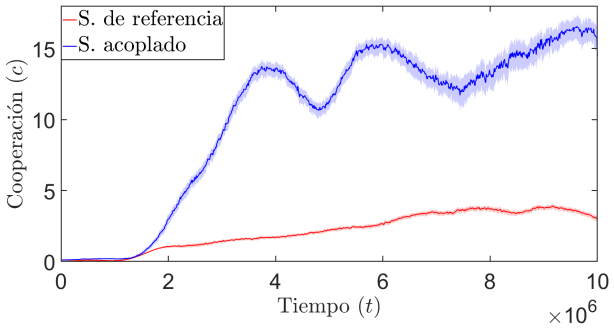


Figura 4-9: Evolución temporal del índice de cooperación en el sistema de diez agentes. La línea sólida es la media muestral obtenida de los 30 experimentos, y las barras de error (sombreado) muestran un intervalo de confianza del 99.5 %.

La dinámica de la información en el sistema se puede apreciar en la figura 4-10, en la que se muestra la evolución temporal de la entropía promedio y del estimado

de la información mutua promedio.

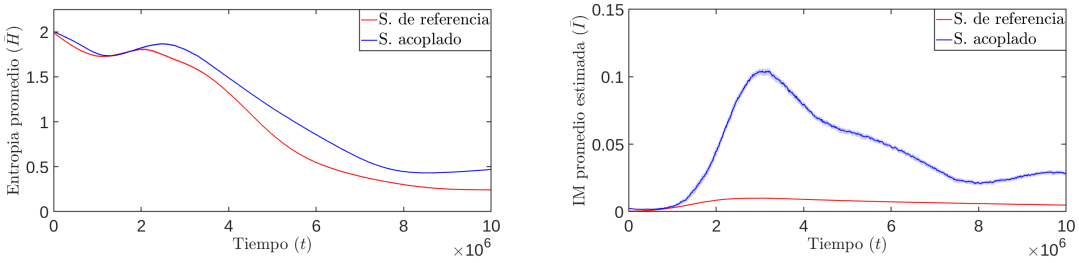


Figura 4-10: Evolución temporal de la información en el sistema de diez agentes. La línea sólida es la media muestral obtenida de los 30 experimentos, y las barras de error (sombreado) muestran un intervalo de confianza del 99.5 %.

La entropía desciende inicialmente cuando las políticas convergen a la sobre-explotación, para luego ascender una vez las condiciones del entorno cambian al superarse la capacidad de regeneración del recurso. En adelante la entropía desciende, más lentamente en el caso del sistema acoplado, obedeciendo a la convergencia de las políticas.

La información mutua promedio en la etapa de aprendizaje de estrategias de sobre-explotación es casi nula, indicando la independencia de los agentes. Una vez empieza la competencia por el recurso esta crece en ambos sistemas, a un ritmo más acelerado en el sistema acoplado y alcanzando valores superiores a los del sistema de referencia. El estimado de la información mutua promedio alcanza un valor máximo de $\bar{I} = 0{,}1043$ en el instante $t = 3{,}06 \times 10^6$, coincidiendo con el aumento de la velocidad de crecimiento de la sostenibilidad, y luego desciende siguiendo la reducción de la entropía de las políticas.

### Resultados del entrenamiento

La caracterización de los resultados del entrenamiento del sistema de diez agentes se describe en la figura 4-11 y en la tabla 4.7. La figura 4-11 muestra los histogramas de los cuatro índices de desempeño en el primer (columna izquierda) y último (columna derecha) episodio para el sistema acoplado y el sistema de referencia, y la tabla 4.7 compara la media y varianza de los índices de desempeño y del índice de cooperación

para los dos sistemas al comienzo y final del entrenamiento.

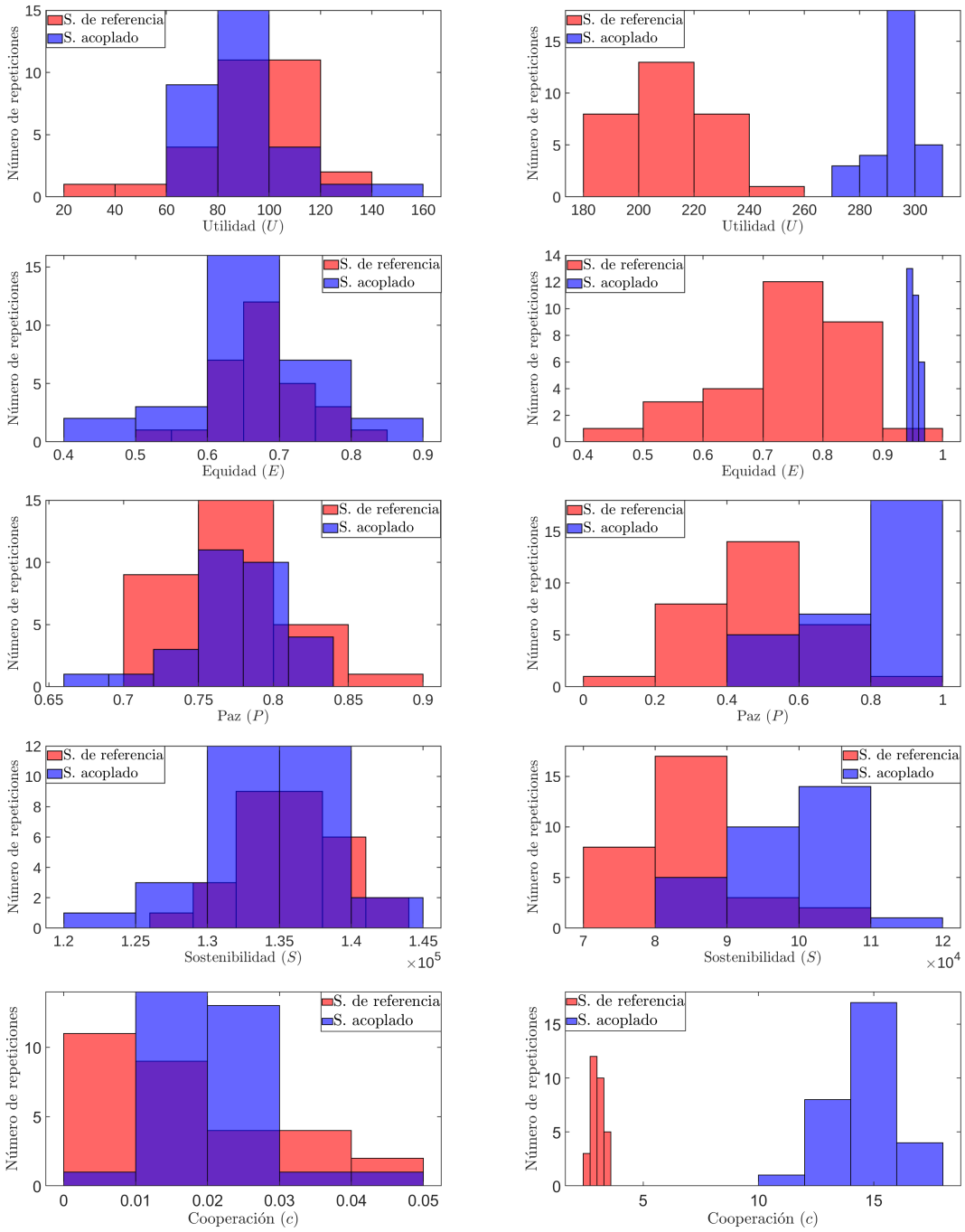


Figura 4-11: Histogramas de los índices de desempeño y del índice de cooperación en el primer episodio (izquierda) y último episodio (derecha) del entrenamiento en el sistema de diez agentes.

Tanto en los histogramas de la figura 4-11 como en la tabla 4.7 se observa que en el sistema de referencia la media de la distribución aumenta con el proceso de

| | | Sistema de referencia | | Sistema acoplado | |
|---|---|---|---|---|---|
| | Índice | Media | Varianza | Media | Varianza |
| Inicial | Utilidad | 93.2728 | 358.9156 | 90.0706 | 321.3627 |
| | Equidad | 0.6778 | 0.0040 | 0.6618 | 0.0075 |
| | Paz | 0.7732 | 0.0016 | 0.7749 | 0.0011 |
| | Sostenibilidad | $1{,}3560 \times 10^5$ | $1{,}2317 \times 10^7$ | $1{,}3472 \times 10^5$ | $1{,}7866 \times 10^7$ |
| | Cooperación | 0.0170 | 0.0001 | 0.0096 | $1{,}1727 \times 10^{-}5$ |
| Final | Utilidad | 211.8189 | 277.6377 | **293.4253** | **52.8959** |
| | Equidad | 0.7464 | 0.0125 | **0.9530** | **0.0001** |
| | Paz | 0.5085 | **0.0266** | **0.8157** | 0.0284 |
| | Sostenibilidad | $8{,}4526 \times 10^4$ | $6{,}0339 \times 10^7$ | $\mathbf{9{,}8540 \times 10^4}$ | $\mathbf{5{,}8366 \times 10^7}$ |
| | Cooperación | 2.9994 | **0.0624** | **14.5037** | 1.9067 |

Tabla 4.7: Comparación de los índices de desempeño y del índice de cooperación entre el sistema acoplado y el sistema de referencia al inicio y final del entrenamiento en el sistema de diez agentes.

entrenamiento, salvo en el índice de paz y el índice de sostenibilidad, y la varianza se reduce para el índice de utilidad. En el sistema acoplado la media aumenta para todos los índices salvo el índice de sostenibilidad, y la varianza se reduce para los índices de utilidad y equidad. Considerando la utilidad como la medida de desempeño, y según la definición de aprendizaje dada en la sección 2.1, se puede decir que el sistema multi-agente ha aprendido en ambos casos.

La diferencia de las distribuciones de los índices de desempeño entre el sistema de referencia y el sistema acoplado en el último episodio muestra al igual que en el caso de dos agentes que el sistema acoplado supera en desempeño al sistema de referencia, obteniendo una media superior en todos los índices, como se indica con los valores en negrilla en la tabla 4.6, y una menor varianza en los índices de utilidad, equidad, y sostenibilidad.

# Capítulo 5

# Conclusiones y trabajo futuro

Se diseñó un sistema multi-agente neuronal entrenado mediante aprendizaje profundo por refuerzo para maximizar sus ganancias y la información mutua entre las acciones de sus agentes. El algoritmo de entrenamiento propuesto para el diseño del sistema es descentralizado, puede ser usado para poblaciones de tamaño variable en el tiempo, y su complejidad computacional es independiente del tamaño de la población, dependiendo únicamente de la dimensión de las observaciones del agente.

El sistema fue aplicado al juego de la cosecha, un dilema social secuencial, en dos configuraciones del entorno y del tamaño de la población de agentes, y su desempeño fue evaluado según los índices de utilidad, equidad, paz y sostenibilidad propuestos en [40] y un índice de cooperación propuesto en este trabajo. El desempeño del sistema diseñado fue comparado con el de un sistema de referencia entrenado sin la maximización de la información mutua, obteniéndose que el primero superó consistentemente al sistema de referencia según todos los índices.

Del análisis de las dinámicas temporales de los índices de desempeño y de la información en el sistema obtenidas experimentalmente se observó que:

- El sistema diseñado con el algoritmo de maximización de la información mutua y el sistema de referencia pasan por una fase inicial caracterizada por una baja información mutua promedio, en la que los agentes aprenden estrategias de sobre-explotación. Una vez la velocidad de explotación del recurso supera su

capacidad de regeneración las políticas de sobre-explotación se vuelven obsoletas y el entorno se vuelve competitivo, lo que resulta en el aumento de la entropía de las políticas, y la transición de los agentes a nuevas estrategias.

- Ambos sistemas pasan por una segunda fase que se caracteriza por el descenso de la entropía a medida que las políticas convergen, y por el crecimiento de la información mutua, obedeciendo a la necesidad de los agentes de reconocer a sus pares al encontrarse ahora en un entorno competitivo.

- El sistema diseñado con el algoritmo de maximización de la información mutua presenta una dinámica más lenta que el sistema de referencia. Esto probablemente debido a que la maximización de la información mutua también incentiva la maximización de la entropía, lo que hace más lenta la convergencia.

- El sistema de referencia converge a una estrategia basada en reducir la población efectiva mediante el conflicto entre agentes, posibilitando la regeneración del recurso, pero produciendo distribuciones desiguales del recurso en la población y utilidades subóptimas.

- En el sistema diseñado con el algoritmo de maximización de la información mutua inicialmente se da la misma estrategia competitiva, sin embargo la información mutua crece a niveles superiores que en el sistema de referencia, indicando acciones conjuntas más coordinadas, lo que se manifiesta en recursos distribuidos más equitativamente y eventualmente en la transición a una estrategia cooperativa, obteniéndose una utilidad mayor que en el sistema de referencia en concordancia con la primera desigualdad de los dilemas sociales.

- En el sistema acoplado el punto máximo de la información mutua coincidió con puntos de transición de fase en las dinámicas de los índices de paz, en el sistema de dos agentes, y sostenibilidad, en el sistema de diez agentes, indicando transiciones a estrategias cooperativas.

Según lo cual se concluye que:

- Las dinámicas de la información en el sistema permiten caracterizar las transiciones entre estrategias competitivas y cooperativas.

- La maximización de la información mutua entre agentes en un problema con las características de un dilema social secuencial es un factor causal en la emergencia de la cooperación, y por tanto, su inclusión como objetivo en un algoritmo de entrenamiento es una herramienta para entrenar sistemas multi-agente que se auto-organicen para cooperar.

## 5.1. Aportes originales

- Se propuso un algoritmo de entrenamiento descentralizado y escalable a poblaciones de cualquier tamaño, y variables en el tiempo, para el entrenamiento de sistemas multi-agente neuronales que se auto-organizan para coordinar, según se mide por el índice de información mutua entre sus acciones.

- Se propuso por primera vez, a conocimiento del autor, la inclusión de una representación diferenciable de la información mutua como función objetivo en un problema de aprendizaje por refuerzo.

- Se planteó un índice de cooperación para sistemas multi-agente que tengan un índice de desempeño definido.

## 5.2. Trabajos futuros

- Evaluar el sistema con poblaciones variables en el tiempo, y determinar su robustez a la inserción de agentes heterogéneos.

- Caracterizar el juego de la cosecha de modo que se conozca la utilidad óptima.

- Caracterizar la sensibilidad del sistema a la variación de parámetros (arquitectura de las redes neuronales, tasas de aprendizaje, pesos de las funciones objetivo, etc.) y determinar el mejor conjunto de parámetros.

- Caracterizar la influencia de los elementos entrenados e inicializados offline (sensor, codificador auxiliar y memoria) en la dinámica del entrenamiento online.

- Evaluar el sistema multi-agente modificando la ecuación 3.24 de la forma:

$$J = \frac{J_\pi \, J_F}{H(\pi)} + w_H H(\pi) \tag{5.1}$$

  , de modo que se maximice directamente el índice de cooperación propuesto en la sección 3.3.

- Caracterizar y comparar las señales generadas por el sistema sistema diseñado con el algoritmo de maximización de la información mutua y el sistema de referencia mediante análisis lineal (autocorrelación, espectro de densidad de potencia, etc.) y no lineal (información mutua promedio, espacio de fase, exponentes de Lyapunov, análisis fractal, etc.).

- Aplicar el sistema multi-agente neuronal diseñado a otros dilemas sociales secuenciales y a problemas que no sean dilemas sociales. Sería de particular interés observar el resultado en problemas cuya solución óptima requiere agentes competitivos.

- Aplicar el sistema multi-agente neuronal diseñado a problemas prácticos de ingeniería. Posibles escenarios de aplicación son las redes de radio cognitiva y las flotas de vehículos autónomos.

# Referencias


[1] A., Krizhevsky, I. Sutskever y G. Hinton: *ImageNet classification with deep convolutional neural networks*. Advances in Neural Information Processing Systems, 2012.

[2] Wu, et al: *Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation*. arXiv preprint arXiv:1609.08144, 2016.

[3] van den Oord, et al: *Wavenet: A generative model for raw audio*. arXiv preprint arXiv:1609.03499, 2016.

[4] Silver, D., et al: *Mastering the game of go with deep neural networks and tree search*. Nature 529, 484–489, 2016.

[5] Goodfellow, Ian, Yoshua Bengio y Aaron Courville: *Deep Learning*. `http://www.deeplearningbook.org`, Libro en preparación, MIT Press, 2016.

[6] Hernandez-Leal, P., B. Kartal y M. E. Taylor: *Is multiagent deep reinforcement learning the answer or the question? A brief survey*. ArXiv e-prints, Octubre 2018.

[7] Shoham, Yoav y Kevin Leyton-Brown: *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge University Press, 2008.

[8] Dawes, R M: *Social Dilemmas*. Annual Review of Psychology, 31(1):169–193, 1980. `https://doi.org/10.1146/annurev.ps.31.020180.001125`.

[9] Marcela Mejia, Néstor Peña, Jose L. Muñoz, Oscar Esparza y Marco Alzate: *DECADE: Distributed Emergent Cooperation through ADaptive Evolution in mobile ad hoc networks*. Ad Hoc Networks, 10(7):1379 – 1398, 2012, ISSN 1570-8705. `http://www.sciencedirect.com/science/article/pii/S1570870512000686`.

[10] Liviu Panait y Sean Luke: *Cooperative Multi-Agent Learning: The State of the Art*. Springer, Autonomous Agents and Multi-Agent Systems, 11, 387–434, 2005, 2005.

[11] Sugiyama, Masashi y Motoaki Kawanabe: *Machine Learning in Non-Stationary Environments: Introduction to Covariate Shift Adaptation*. The MIT Press, 2012, ISBN 0262017091, 9780262017091.

[12] Tsitsiklis, J. N. y B. Van Roy: *An analysis of temporal-difference learning with function approximation*. IEEE Transactions on Automatic Control, 42(5):674–690, May 1997, ISSN 0018-9286.

[13] Lowe, Ryan, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel y Igor Mordatch: *Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments*. CoRR, abs/1706.02275, 2017. http://arxiv.org/abs/1706.02275.

[14] Foerster, Jakob N., Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli y Shimon Whiteson: *Counterfactual Multi-Agent Policy Gradients*. CoRR, abs/1705.08926, 2017. http://arxiv.org/abs/1705.08926.

[15] Kok, Jelle R. y Nikos Vlassis: *Collaborative Multiagent Reinforcement Learning by Payoff Propagation*. JOURNAL OF MACHINE LEARNING RESEARCH, 7:1789–1828, 2006.

[16] Peng, Peng, Quan Yuan, Ying Wen, Yaodong Yang, Zhenkun Tang, Haitao Long y Jun Wang: *Multiagent Bidirectionally-Coordinated Nets for Learning to Play StarCraft Combat Games*. CoRR, abs/1703.10069, 2017. http://arxiv.org/abs/1703.10069.

[17] Foerster, Jakob N., Richard Y. Chen, Maruan Al-Shedivat, Shimon Whiteson, Pieter Abbeel y Igor Mordatch: *Learning with Opponent-Learning Awareness*. CoRR, abs/1709.04326, 2017. http://arxiv.org/abs/1709.04326.

[18] Omidshafiei, S., J. Pazis, C. Amato, J. P. How y J. Vian: *Deep Decentralized Multi-task Multi-Agent Reinforcement Learning under Partial Observability*. ArXiv e-prints, Marzo 2017.

[19] Cover, Thomas M. y Joy A. Thomas: *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, New York, NY, USA, 2006, ISBN 0471241954.

[20] Kolchinsky, Artemy y David H. Wolpert: *Semantic information, autonomous agency, and nonequilibrium statistical physics*. arXiv e-prints, página arXiv:1806.08053, Jun 2018.

[21] Rovelli, Carlo: *Meaning = Information + Evolution*. arXiv e-prints, página arXiv:1611.02420, Nov 2016.

[22] E Freer, C: *Causal Entropic Forces*. Physical review letters, 110:168702, Abril 2013.

[23] Miguel Melgarejo y Nelson Obregon: *Information Dynamics in Urban Crime*. Entropy, 20(11), 2018, ISSN 1099-4300. http://www.mdpi.com/1099-4300/20/11/874.

[24] Dolgonosov, B.M. y V.I. Naidenov: *An informational framework for human population dynamics*. Ecological Modelling, 198(3):375 – 386, 2006, ISSN 0304-3800. http://www.sciencedirect.com/science/article/pii/S0304380006002390.

[25] Griffith, Virgil y Christof Koch: *Quantifying synergistic mutual information*. arXiv e-prints, página arXiv:1205.4265, May 2012.

[26] Biehl, Martin: *Formal approaches to a definition of agents*. CoRR, abs/1704.02716, 2017. http://arxiv.org/abs/1704.02716.

[27] Mitchell, Thomas M.: *Machine Learning*. McGraw-Hill, Inc., New York, NY, USA, 1ª edición, 1997, ISBN 0070428077, 9780070428072.

[28] Mnih, V., et al: *Human-level control through deep reinforcement learning*. Nature 518, 529–533, 2015.

[29] Sutton, R. y A. Barto: *Reinforcement Learning: An Introduction*. The MIT Press, 2012.

[30] Kaelbling, L, M Littman y A Moore: *Reinforcement Learning A Survey*. Journal of Artificial Intelligence Research, 1996.

[31] Gosavi, Abhijit: *Reinforcement Learning: A Tutorial Survey and Recent Advances*. INFORMS Journal on Computing, vol. 21, no. 2, pp. 178–192, 2009.

[32] Schmidhuber, J.: *Deep Learning in Neural Networks: An Overview*. ArXiv e-prints, Abril 2014.

[33] Watkins, J, H C y P Dayan: *Q-learning*. Technical Note, Machine Learning, 1992.

[34] Rummery G, A y M Niranjan: *Online Q Learning using connectionist systems*. Tech. rep. CUED/F-INFENG/TR166, Cambridge University, 1994.

[35] Sutton, R, D McAllester, S Singh y Y Mansour: *Policy Gradient Methods for Reinforcement Learning with Function Approximation*. Advances in Neural Information Processing Systems 12, pág. 1057–1063, 2000.

[36] François-Lavet, Vincent, Peter Henderson, Riashat Islam, Marc G. Bellemare y Joelle Pineau: *An Introduction to Deep Reinforcement Learning*. CoRR, abs/1811.12560, 2018. http://arxiv.org/abs/1811.12560.

[37] Arulkumaran, Deisenroth, Brundage y Bharath: *A Brief Survey of Deep Reinforcement Learning*. IEEE Signal Processing Magazine, special issue on Deep Learning for image understanding (arXiv extended version), 2017.

[38] Macy, Michael W. y Andreas Flache: *Learning dynamics in social dilemmas*. Proceedings of the National Academy of Sciences, 99(suppl 3):7229–7236, 2002, ISSN 0027-8424. https://www.pnas.org/content/99/suppl_3/7229.

[39] Leibo, Joel Z., Vinícius Flores Zambaldi, Marc Lanctot, Janusz Marecki y Thore Graepel: *Multi-agent Reinforcement Learning in Sequential Social Dilemmas*. CoRR, abs/1702.03037, 2017. http://arxiv.org/abs/1702.03037.

[40] Pérolat, Julien, Joel Z. Leibo, Vinícius Flores Zambaldi, Charles Beattie, Karl Tuyls y Thore Graepel: *A multi-agent reinforcement learning model of common-pool resource appropriation*. CoRR, abs/1707.06600, 2017. http://arxiv.org/abs/1707.06600.

[41] Belghazi, Ishmael, Sai Rajeswar, Aristide Baratin, R. Devon Hjelm y Aaron C. Courville: *MINE: Mutual Information Neural Estimation*. CoRR, abs/1801.04062, 2018. http://arxiv.org/abs/1801.04062.

[42] Devon Hjelm, R, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler y Yoshua Bengio: *Learning deep representations by mutual information estimation and maximization*. arXiv e-prints, página arXiv:1808.06670, Aug 2018.

[43] Nowozin, Sebastian, Botond Cseke y Ryota Tomioka: *f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization*. arXiv e-prints, página arXiv:1606.00709, Jun 2016.

[44] Ottino, J: *Engineering Complex Systems*. Nature, 427:399, Febrero 2004.

[45] Schrauwen, Benjamin, David Verstraeten y Jan Van Campenhout: *An overview of reservoir computing: theory, applications and implementations*. En *Proceedings of the 15th European Symposium on Artificial Neural Networks. p. 471-482 2007*, páginas 471–482, 2007. http://dx.doi.org/1854/11063.

[46] Gruslys, Audrunas, Remi Munos, Ivo Danihelka, Marc Lanctot y Alex Graves: *Memory-Efficient Backpropagation Through Time*. En Lee, D. D., M. Sugiyama, U. V. Luxburg, I. Guyon y R. Garnett (editores): *Advances in Neural Information Processing Systems 29*, páginas 4125–4133. Curran Associates, Inc., 2016. http://papers.nips.cc/paper/6221-memory-efficient-backpropagation-through-time.pdf.

[47] Gallicchio, Claudio: *Chasing the Echo State Property*. CoRR, abs/1811.10892, 2018. http://arxiv.org/abs/1811.10892.

[48] Barančok, Peter y Igor Farkaš: *Memory Capacity of Input-Driven Echo State Networks at the Edge of Chaos*. En Wermter, Stefan, Cornelius Weber, Włodzisław Duch, Timo Honkela, Petia Koprinkova-Hristova, Sven Magg, Günther Palm y Alessandro E. P. Villa (editores): *Artificial Neural Networks and Machine Learning – ICANN 2014*, páginas 41–48, Cham, 2014. Springer International Publishing, ISBN 978-3-319-11179-7.

[49] Bakker, Bram: *Reinforcement Learning with LSTM in Non-Markovian Tasks with Long-Term Dependencies*. Technical report, Leiden University, 2001.

[50] Schulman, Wolski, Dhariwal, Radford y Klimov: *Proximal Policy Optimization Algorithms*. arXiv preprint arXiv:1707.06347, 2017.

[51] Ceriani, Lidia y Paolo Verme: *The origins of the Gini index: extracts from VariabilitA e MutabilitA (1912) by Corrado Gini*. Journal of Economic Inequality - J ECON INEQUAL, 10:1–23, Septiembre 2012.

[52] Kingma, Diederik P. y Jimmy Ba: *Adam: A Method for Stochastic Optimization*, 2014. `http://arxiv.org/abs/1412.6980`, cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015.

[53] Chang, Hanten y Katsuya Futagami: *Convolutional Reservoir Computing for World Models*. CoRR, abs/1907.08040, 2019. `http://arxiv.org/abs/1907.08040`.