INSTITUTO TECNOLÓGICO DE COSTA RICA
ESCUELA INGENIERÍA EN COMPUTACIÓN
PROGRAMA DE MAESTRÍA

Entrenamiento de Redes Neuronales para Agente Especializado de Robocopa

Tesis para optar al grado de
Magister Scientiae en Computación

Christian Calvo Masís

Cartago, Costa Rica
Junio 2013

Entrenamiento de Redes Neuronales para Agente Especializado de Robocopa

Resumen

El aprendizaje es un área de la inteligencia artificial de especial interés para un grupo grande de investigadores. Una de las técnicas que ha tomado auge en los últimos años es la de aprendizaje mediante redes neuronales, y se han desarrollado diversos tipos de redes para distintos tipos de problemas. Una de las redes más utilizada es el modelo de redes multicapa (multilayer perceptrons) entrenadas con el algoritmo de retropropagación. Sobre ellas se han desarrollado propuestas de modificaciones sea en los algotimos de aprendizaje o bien en la cantidad de capas y sobre cómo lograr un correcto aprendizaje. Los juegos por su parte, son un dominio idóneo para el ensayo de técnicas de inteligencia artificial, incluyendo las redes neuronales, ya que con ellos se puede probar, dentro de ambientes controlados, la eficiencia de las distintas técnicas. Un ambiente de investigación particular muy utilizado por la complejidad y gama de problemas de IA que se puede ensayar e investigar en él, es el de Robocopa (Robocup) donde agentes inteligentes, como robots o simulación, compiten por equipos en juegos de fútbol. La presente investigación se centra en la especialización de jugadores de Robocopa, mediante el diseño y uso de redes neuronales de retropropagación y el diseño y depuración de sus conjuntos de entrenamiento. Primeramente se provee un resumen de literatura pertinente sobre agentes inteligentes, Robocopa, Redes Neuronales y algunas aplicaciones en Robocopa, mostrando también un compendio de tácticas de fútbol, recomendadas por entrenadores. Seguidamente se muestran los resultados de la investigación diferenciando los jugadores básicamente por su comportamiento (que está dado por una red neuronal diferente para cada tipo de jugador definido), además se muestran los resultados del entrenamiento de dichas redes. Los experimentos se realizan en herramientas básicas y en un ambiente limitado haciendo énfasis en los aspectos de diseño de las redes y sus conjuntos de entrenamiento. A partir de los resultados se extraen conclusiones y recomendaciones relativas a los objetivos de la investigación, luego se formulan un conjunto de propuestas para ser desarrolladas en futuras investigaciones.

Palabras clave: Inteligencia Artificial, Aprendizaje, Redes Neuronales, Retropropagación, Entrenamiento, Robocopa.

Neural Networks Training for an Specialized Robocup Agent

Abstract

An area of Artificial Intelligence (AI) that is of great interest for a numerous group of researchers is learning. One of the most popular techniques developed in the last years has been learning through artificial neural networks. Different types of these networks have been developed for a diversity of problems. One of the most common models is multilayer perceptrons trained with backpropagation algorithm. Some modification proposals have been developed either in the algorithms of learning or in the number of layers and how to achieve the right learning. Games are the perfect environment to test rehearse the different AI techniques (including artificial neural networks) since the level of efficiency of such techniques can be tested in a controlled environment. A particular research environment frequently used for its complexity and the amount of AI problem that can be rehearse and research on is the Robocup where intelligent agents like robots or simulations team up and compete in soccer games. This research paper focuses on the specialization of the players in Robocup by designing and using different backpropagation artificial neural networks as well as the design and depuration of its training sets. First of all, a review of the literature regarding intelligent agents, the Robocup, artificial neural network, and some applications in the Robocup is provided along with a summary of certain soccer tactics suggested by soccer coaches. Then, the results of the research are displayed establishing a difference among the players based basically on their behavior (given by an artificial neural network different to each type of player); besides, results of the training of such networks is presented. The experiments are carried out by using simple tolls and in a limited environment emphasising the design aspects and their training sets. From the results obtained some conclusions and recommendations are drawn. Finally a proposal to be developed in future research is proposed.

Keywords: Artificial Intelligence, Learning, Neural Networks, Backpropagation, Training, Robocup.

TEC | Tecnológico de Costa Rica

Programa de Maestría
en Computación
Teléfono: 2550-2402

APROBACIÓN DE LA TESIS

"Entrenamiento de Redes Neuronales para Agente Especializado de Robocopa"

TRIBUNAL EXAMINADOR

Ph.D. José Castro Mora
Profesor Asesor

Ph.D. José Enrique Araya Monge
Profesor Lector

MBA Luis Naranjo Zeledón
Profesional Externo

M.Sc. Luis Carlos Loaiza Canet
Coordinador a.i. del Programa
de Maestría en Computación

Junio, 2013

A mi esposa Grettel, a mis padres y a María Guadalupe

Agradezco a José Castro, Ph.D. por la guía brindada en el desarrollo de esta investigación, a Luis Naranjo Zeledón, M. Phil y José Enrique Araya, Ph.D. por sus oportunos comentarios.

“Cuándo verá la luz lo que has esperado,
cuándo será vivido lo soñado,
cuando suceda más nos brillará el sol.”
J. Capmany

Índice General

1 . Introducción y Antecedentes

Una de las ramas de la computación que ha interesado más a los investigadores en los últimos años es la Inteligencia Artificial (IA). Proyectos como el DARPA Robotics Challenge [DRC13] en el que se busca crear robots que puedan ejecutar funciones peligrosas para los humanos en situaciones de desastres naturales o provocados por el ser humano o bien el DARPA Grand Challenge que “es una carrera de coches sin piloto patrocinada por la estadounidense Agencia de Investigación de Proyectos Avanzados de Defensa” [ECOI11], muestran el creciente interés en el área.

La IA a su vez, tiene una amplia gama de ramas de investigación, tales como aprendizaje, razonamiento automático, planificación, visión, reconocimiento de patrones, reconocimiento de lenguaje, etc. en las cuales se utilizan muy diversas técnicas, tales como árboles de decisión, sistemas expertos, redes neuronales, etc. Cada rama tiene también muy diversas aplicaciones.

Desde mediados de los ochenta, algunos investigadores empezaron a promover la idea de los agentes inteligentes y hacia mediados de los noventa se empezó a tomar mayor interés en éstos [SEDE04], este interés buscaba crear bases teóricas sólidas sobre cómo funcionan e interactúan los sistemas inteligientes [WOJE95]. Hay muchas definiciones de agentes inteligentes, aquí utilizamos como referencia la de [JUB00]: Los Agentes Inteligentes son programas reactivos, proactivos y sociales, capaces de tomar decisiones de

manera autónoma a la vez que pueden colaborar con otros. El interés hacia los agentes ha crecido dado a la diversidad de ambientes en que se pueden desempeñar [RUNO10] y como lo apuntan Serenko y Detlor [SEDE04] se han aplicado en diversas situaciones para ayudar en tareas complejas o repetitivas, como “Web guides, personal advisors, shopping assistants, virtual educators, and entertainers” [SEDE04]. Dichos agentes pueden implementar diversas técnicas de IA.

Los ambientes con que interactúa y se encuentra inmerso un agente pueden ser muy variados y entonces resulta útil categorizar a estos con respecto a su complejidad. Russel y Norvig [RUNO10] sugieren categorizar la autonomía requerida dependiendo si el ambiente es perceptible parcialmente o por completo, si la tarea se puede solventar por un programa común o por un agente, si el contexto con que interactúa contiene otros agentes (multiagente), si el ambiente es determinístico o estocástico, etc.

Un tipo de contexto muy llamativo en el cual se pueden implementar técnicas de IA son los juegos. Kitano et al [KIAS97] hacen mención al ajedrez y al fútbol de robocopa, ya que en estos ambientes controlados se pueden ensayar las técnicas de IA y medir su impacto, características, y utilidad. En particular en el contexto de Robocopa, en el cual se implementan agentes que juegan fútbol, se requiere de prácticamente todas las técnicas de IA de las cuales se pueda echar mano, ya que el ambiente necesita de colaboración entre agentes, involucra competencia, hay información parcial con situaciones que cambian rápidamente,

etc.

En un juego de fútbol, varios jugadores, cada uno con sus propias destrezas, buscan obtener un objetivo común: anotar y evitar que les anoten. Cada jugador actúa independientemente, pero siempre colaborando con los demás para obtener la meta. Este problema es especialmente particular, el comportamiento del equipo es un comportamiento emergente, resultado de la interacción y cooperación de los agentes individuales; sin embargo, se pueden buscar acciones netamente colectivas mediante jugadas planificadas, donde el comportamiento del individuo va a estar sujeto al plan grupal.

Son diversas las técnicas de IA que se pueden incluir en la creación de jugadores de fútbol. Esta investigación busca abordar principalmente el uso de las redes neuronales de retropropagación, para hacer que el jugador aprenda a tomar ciertas decisiones.

Si bien se ha comprobado en estudios, como los realizados por José Castro [CAST00] y José Helo [HELO96] y muchos otros en sus referencias, que el algoritmo de retropropagación tiene algunos problemas con respecto a otros algoritmos, como lo son los mínimos locales [OTSA05], el problema de parálisis en el aprendizaje de la red, su lentitud en su tiempo de convergencia, como los señalados por Otair y Salamed [OTSA05] y Carpenter et al [CAGR91]; también es cierto que ha sido utilizado exitosamente en áreas como el reconocimiento de patrones [RUSS05] en problemas críticos. Russel [RUSS05] apunta que Stubbs logró que redes de retropropagación predijeran la frecuencia de reacciones

adversas para un tipo de fármacos.

Recientemente Hinton [HINT06] ha liderado un auge en el uso y conocimiento del algoritmo, problemas que tenía como su dificultad para entrenar redes de varias capas, han sido superados utilizando un pre-entrenamiento no supervisado por capa [HINT07] donde se crean conexiones top-down para permitir que las redes construyan detectores de características (feature detectors) que permitan acelerar el aprendizaje, y limitantes en su capacidad de generalización han resultado ser solo problemas de insuficiencia de datos que no se podían resolver hasta ahora que se vive en una época de una gran explosión de información.

Por estos motivos, para efectos de esta investigación se considera que el algoritmo puede adaptarse al problema en estudio y que además es conveniente por las diferentes herramientas que lo implementan, su simplicidad y la amplia literatura disponible en cuanto a posibles mejoras o modificaciones, como las apuntadas, de nuevo, por Hinton [HINT06]

Para diseñar una red neuronal para un jugador de fútbol, se debe tener un marco de referencia del ambiente en el cual ese jugador existiría, esto pues, si es un simulador, se debe conocer cuál información tiene disponible y cuáles acciones el jugador podría realizar. En esta investigación se utilizará el ambiente de Robocopa, un ambiente que es ideal para esta tarea, pues ha sido ampliamente utilizado para realizar equipos de fútbol empleando diferentes técnicas.

Kitano et al [KIAS97] recalcan algunas cualidades importantes de Robocopa

que hacen que sea de especial interés para esta investigación:

> La Robocopa está diseñada para manejar la complejidad del mundo real, aunque en un ambiente limitado, mientras que mantiene un tamaño y costo de investigación razonable. La Robocopa ofrece una tarea de investigación integrada cubriendo amplias áreas de la IA y robótica, incluyendo fusión de sensores en tiempo real, comportamiento reactivo, adquisición de estrategia, aprendizaje, planificación en tiempo real, sistemas multiagente, reconocimiento de contexto, visión, toma de decisiones estratégicas, control motor y control inteligente de robots[1].

La Robocopa es un proyecto conjunto internacional para promover el desarrollo de las técnicas de inteligencia artificial (IA), robótica y áreas afines; es un intento para fomentar la investigación en IA y robots inteligentes proveyendo un problema estándar donde muchas técnicas de IA pueden ser integradas [CHFO03].

El tipo de agente jugador de fútbol para el cual se diseñarán las redes es el que se ejecuta en la liga de simuladores de Robocopa, la cual está basada en un componente principal: el “Soccer Server” (servidor) que es el simulador del sistema físico. El ambiente de simulación de robocopa cuenta con otros componentes, los cuales son el “soccer monitor”, monitor para visualizar los juegos, y los “soccer players”, agentes jugadores que interactúan con el servidor.

1 Traducción libre del autor al texto: “RoboCup is designed to handle real-world complexities, although in a limited world, while it maintains an affordable size and research cost. RoboCup offers an integrated research task covering the broad areas of AI and robotics, including real-time sensor fusion, reactive behavior, strategy acquisition, learning, real-time planning, multiagent systems, context recognition, vision, strategic decision making, motor control, and intelligent robot control” [KIAS97].

La ventaja del proyecto del simulador es que no es necesario preocuparse por aspectos físicos como reconocimiento de objetos, comunicación, problemas de hardware, etc; los investigadores se pueden concentrar es aspectos como la colaboración y aprendizaje. [CHFO03]

Algunos estudios que han aplicado las redes neuronales de retropropagación a situaciones de Robocopa son las de Kalyviotis y Hu [KAHU02] que modelan la toma de decisiones para la jugada de saque de puerta, y la de David Abraham [ABRA01] donde se utiliza una red para una situación muy simple de decidir si un jugador pasa la bola o remata a marco.

Hossein Dezfoulian M. et al [HOKA05] utilizaron con éxito las redes neuronales de retropropagación para que el jugador decida la posibilidad de poder anotar si remata a marco y cuál es la mejor posición del marco para rematar.

2 . Planteamiento del Problema

En esta investigación se busca proponer diseños de redes neuronales apropiados para que un jugador de Robocopa pueda tomar sus decisiones, así como crear, evaluar y proponer conjuntos de entrenamiento apropiados para lograr un buen desempeño de las redes; así mismo, se evaluarán diversas técnicas de entrenamiento concluyendo cuáles son más aptas de acuerdo con las pruebas realizadas.

Este es un primer paso en una línea de investigación para incorporar diversas técnicas de IA en agentes jugadores de fútbol de Robocopa, con la limitante de que no se posee información estadística para comparar la aproximación utilizada, basada en redes neuronales, contra otras posibles que utilcen diversas técnicas.

3 . Justificación del Problema

Esta investigación ayudará a los futuros investigadores en el área de Robocopa que deseen implementar redes neuronales para lograr un proceso de toma de decisiones eficientes en los jugadores. Además sentará las bases para investigaciones futuras donde se podrían implementar diferentes variantes del algoritmo de retropropagación para lograr el mismo fin, o bien otros algoritmos de entrenamiento de redes neuronales.

Investigaciones anteriores [ABRA01] [KAHU02] [HOKA05] han mostrado que pese a que las Redes Neuronales son generalmente usadas en problemas de clasificación, un problema de toma de decisiones se puede ver como un tipo de clasificación de manera que las Redes Neuronales sean una alternativa válida y eficiente, permitiendo que aproximaciones similares se utilicen en otros procesos de toma de decisiones.

4 . Objetivos

4.1 Objetivo General

Diseñar redes neuronales para la toma de decisiones de un jugador de simulación de Robocopa, así como los conjuntos de entrenamiento adecuados, proponiendo un método apto para realizar el entrenamiento.

4.2 Objetivos Específicos

1. Comprender el ambiente de simulación de Robocopa y las decisiones importantes que debe tomar un agente.
2. Definir la especialización básica necesaria para diferenciar el comportamiento de agentes de Robocopa.
3. Identificar escenarios que provean ejemplos de decisiones importantes para un jugador.
4. Definir las redes neuronales necesarias para especializar el comprotamiento de los jugadores.
5. Definir la estructura de cada red neuronal.
6. Crear los conjuntos de entrenamiento que permitan un buen desempeño de cada red.
7. Evaluar las redes creadas y entrenadas extrayendo conclusiones sobre los conjuntos de entrenamiento, diseño de la red y técnicas utilizadas.

5 . Hipótesis o Supuestos

Como parte del proceso de investigación se parte de los siguientes supuestos:

- Algunas decisiones críticas que debe tomar un jugador de fútbol de Robocopa pueden ser modeladas con redes neuronales de retropropagación.
- Las redes neuronales de retropropagación se pueden diseñar y entrenar para responder a situaciones de juego donde las entradas a la red son pocas y el conjunto de entrenamiento pequeño.
- El diseño del conjunto de entrenamiento es crítico para entrenar la red en especial cuando la totalidad de casos no es muy amplia.

6 . Alcances y delimitaciones de la investigación

Esta investigación se muestra delimitada por los siguientes parámetros:

- Se centrará en la realización y entrenamiento de redes neuronales de retropropagación para modelar la toma de decisiones de un agente de simulación de Robocopa, con especial énfasis en el diseño del conjunto de entrenamiento.
- No se entrará en la programación de un agente de simulación de Robocopa completo y competitivo, las pruebas realizadas se limitarán a evaluar el desempeño de las redes propuestas.
- Se buscarán posibilidades de mejora en los conjuntos de entrenamiento y en el procedimiento utilizado para entrenar la red, los resultados se muestran en la sección 8.
- No se dispone de información estadística del desempeño de jugadores de Robocopa que tomen las mismas decisiones modeladas mediante otras técnicas de aprendizaje, lo cual no permitirá elementos de comparación entre la aproximación propuesta y otras posibles.

7 . Revisión de literatura

En esta investigación se busca implementar, mediante el uso de redes neuronales algunos aspectos claves en la toma de decisiones de agentes de simulación de Robocopa, buscando que las acciones realizadas contribuyan a su desempeño individual y a la colaboración efectiva entre los jugadores del equipo.

A continuación se muestra un resumen de literatura consultada que aporta un fundamento teórico para los conceptos desarrollados en la investigación, iniciando por conceptos básicos de agentes que permitan comprender algunos aspectos deseables en jugadores de Robocopa, se brindan los conceptos generales para comprender el ambiente de la Robocopa, luego se presenta un resumen de literatura sobre redes neuronales que orientó el diseño de los experimentos realizados, finalizando con un resumen de las principales estrategias de fútbol que guían las decisiones que las redes neuronales ayudarán a tomar.

7.1 Arquitectura de agentes

Giráldez inicia su descripción de agentes con lo siguiente: “Un agente es una entidad que actúa en representación de otra. Esta definición ha sido empleada como base de diversas, y a veces abusivas y desafortunadas, metáforas en el campo de la informática. No existe una definición ampliamente aceptada por la comunidad que trabaja en SMA [Sistemas Multi Agente] para caracterizar lo que es un agente.” [GIRÁ99].

Franklin y Graesser lo definen como: “Un agente autónomo es un sistema situado dentro de un ambiente, que siente el ambiente y actúa en él, con el tiempo, él busca cumplir su propia agenda y como efecto de esto lo que siente en el futuro”[2][FRGR96].

Dado lo expuesto anteriormente, a continuación se muestra una definición de agente partiendo de las dadas por diversos autores, buscando así generar un concepto bastante amplio y acertado de lo que son, que sirva para guiar el trabajo a realizar en esta investigación.

Los agentes son programas que pueden tomar decisiones por sí mismos para lograr un objetivo utilizando la información disponible del entorno y capaces de reaccionar ante los cambios [ROCA03]. Los agentes realizan este trabajo mediante perceptores y actuadores que les permiten obtener información y actuar en el medio [JUBO00]. Estos agentes pueden también ser capaces de aprender de sus experiencias pasadas para aplicarlas en el futuro [SEDE04]; actúan de acuerdo con su agenda y buscan influir en lo que perciban en el futuro [FRGR96].

Se puede resumir el concepto de agente como todo aquello que pueda percibir y actuar en un entorno, a través de algunos sensores o perceptores, y realiza alguna acción sobre ese mismo medio a través de algunos efectores [JUBO00].

2 Traducción libre del autor al texto: “An autonomous agent is a system situated within and a part of an environment that senses that environment and acts on it, over time, in pursuit of its own agenda and so as to effect what it senses in the future”[FRGR96].

7.1.1 Características generales de agentes

Según Julián et al. [JUBO00] un agente debe tener ciertas características tales como actuar de forma autónoma y flexible en el ambiente. Se entiende como flexible que sea **reactivo** (capaz de responder al entorno), **pro-activo** (intenta cumplir sus propios planes) y **social** (se puede comunicar con otros agentes).

El hecho de que un agente sea autónomo supone que las acciones que realice en el entorno estén de acuerdo con sus percepciones y a su propia agenda [PETR96], realizando las acciones para las cuales fue creado, esto hace que su agenda y las modificaciones que realice en sus metas como resultado de la evaluación del entorno sean intrínsecamente subjetivas, siempre va a buscar su propio beneficio, es por esto que se puede afirmar que los "*agentes no son benevolentes y para cooperar deben recibir algún beneficio individual*"[3] [GRIF05].

Julián et al. [JUBO00] propone algunas características que los agentes deben tener, tales como: actuar siguiendo su propio plan, ejecutarse continuamente, ser autónomos, comunicarse con otros agentes, ser racional (hacer lo correcto de acuerdo con los datos que percibe), actuar de acuerdo con los cambios del entorno, controlar sus propios objetivos (pro-activo), que pueda cambiar su comportamiento de acuerdo con lo que aprende y estar dispuesto a colaborar con otros agentes que no vayan contra sus objetivos.

Wooldridge y Jennings [WOJE95] por su parte, argumentan que utilizar las características de autonomía, sociabilidad, reactividad y proactividad para definir

3 Traducción libre del autor al texto: "agents are not benevolent and to cooperate they must receive some individual benefit"[GRIF05].

un agente provee una visión débil de los mismos, apuntan a la vez que para tener una definición más robusta de agente se debe pensar en características como conocimiento, creencias, intención y obligación, que son nociones mentalistas [MIRA00].

Características como las mencionadas anteriormente pueden ser abordadas mediante técnicas tradicionales de programación o con técnicas de inteligencia artificial. Zarout et al. [ZABO05] apuntan que el principal problema a solucionar es hacer que haya un balance entre la autonomía del agente, es decir, sus propios objetivos y la "*coherencia global del sistema*" [ZABO05].

La sociabilidad, es una de las características más importantes de los agentes, pues es la que permite que se abarquen problemas mayores con mayor eficiencia que si un solo agente o programa trate de abordarlos. Es por eso que según Zarout et al. "las sociedades de agentes parecen ser el candidato más prometedor para el desarrollo del manejo virtual de empresas"[ZABO05] a través de la distribución de las metas, planeación y calendarización de recursos distribuidos y sistemas expertos distribuidos [ZABO05], aspectos en los que la teoría de agentes hereda mucho de lo ya investigado en teoría de procesos paralelos, donde factores como concurrencia, calendarización, comunicación y sincronización son vitales [JAJA92].

Wooldridge y Jennings afirman que si bien el tener una definición de agente puede limitar la visión introduciendo cierto ruido en la producción de los mismos [WOJE95], es conveniente dar un vistazo a diferentes aproximaciones que se han

hecho para deducir algunas características que son convenientes (y otras que no tanto) para los agentes inteligentes. Siempre se debe tener en cuenta que lo que se busca es que los agentes sean “una forma flexible y óptima de gestionar situaciones dinámicas de cambios en las circunstancias, prioridades, recursos y requisitos” [ROCA03].

Wooldridge y Jennings [WOJE95] ya investigaron algunas de las aproximaciones hechas, su trabajo ha sido ampliamente citado, por ejemplo en “*Agents on the Loose: An Overview of Agent Technologies*”[MIRA00], se nota una gran influencia de estos autores [WOJE95] y sus siguientes trabajos. En la siguiente sección se extraen ideas de importancia de Wooldridge y Jennings [WOJE95] con el fin de proveer un marco que guíe el diseño de agente para el cual en esta investigación se crearán las redes neuronales.

7.1.2 Teorías de Agentes

Una de las teorías de agentes que Wooldridge y Jennings [WOJE95] rescatan es la de ver los agentes como sistemas de intenciones, definiendo el agente en términos de actitud y proactitud, así hacen una clasificación de las características en orientadas a la actitud y orientadas a la proacción. Definen entonces que en lo relativo a la actitud, características deseables son las creencias y el conocimiento; relativo a la proacción, los deseos, intenciones, obligaciones, compromiso, elección, etc.

Una conclusión interesante de estos autores [WOJE95] es que la lógica

tradicional no es suficiente para representar la noción de intención, para esto se necesitarían formalismos alternativos. De esta manera, proponen representar las intenciones mediante lenguajes modales o meta-lenguajes que pueden anidar tanto como se necesite.

Para la representación semántica de los agentes, una opción que presentan Wooldridge y Jennings [WOJE95] es la de los Posibles Mundos, en la cual los Posibles Mundos que se pueden dar se representan mediante símbolos y relaciones de accesibilidad. Dicha aproximación no se utiliza en esta investigación dado que como se verá más adelante no se requiere de un formalismo simbólico, pues las decisiones se toman basadas en las redes neuronales que se alimentan de lo que el jugador pueda percibir; sin embargo, sí podría ser de utilidad para abordar el modelaje de jugadas planificadas.

7.1.3 Algunas Arquitecturas de Agentes

Wooldridge y Jennings [WOJE95] retoman una definición de arquitectura de agente dada por Maes, en donde una Arquitectura de Agente especifica como un agente se descompone en módulos y la interacción entre éstos. Esto, junto a los sensores y el estado interno del agente, determina las acciones a tomar y el siguiente estado interno [WOJE95].

Para realizar la implementación de un agente se pueden utilizar algunas de las diferentes arquitecturas resumidas por Wooldridge y Jennings en [WOJE95].

7.1.3.1 Aproximaciones clásicas: Deliberativas

Son alternativas que crean un modelo simbólico del mundo, las decisiones se toman por razonamiento simbólico [WOJE95].

Hay dos problemas básicos que resolver [WOJE95]:

1. Cómo traducir el mundo a símbolos de manera rápida.
2. Cómo representar información de entidades y procesos complejos.

Bajo esta aproximación se han creado los siguientes tipos de agentes [WOJE95]:

Planning Agents

Son sistemas que crean una descripción simbólica del mundo y una meta. Tomando estas descripciones crean un conjunto de acciones para realizarlas [WOJE95].

Máquinas Inteligentes de Resultados

Son sistemas con una biblioteca de planes y representaciones explícitas para las creencias, deseos e intenciones. Tienen un módulo razonador, un analizador de oportunidades, un proceso de filtrado y uno de deliberación [WOJE95].

7.1.3.2 Aproximaciones alternativas: Arquitecturas reactivas

Son esquemas que no se basan en un sistema lógico formal ni un razonamiento simbólico complejo [WOJE95]. Wooldridge y Jennings [WOJE95]

citan propuestas de Brook's para esta aproximación, donde citando a Brook observan que:

1. Actuar inteligentemente se puede alcanzar sin representaciones explícitas.
2. Actuar inteligentemente se puede alcanzar sin razonamiento explícito.
3. La inteligencia es una propiedad emergente en ciertos sistemas complejos.

Brooks basa sus experimentos en lo que llama subsumption architecture, que es una jerarquía de comportamientos (behaviors) de realización de tareas. Cada comportamiento compite por ejercer el control, los de bajo nivel, los más básicos (como evitar obstáculos) tienen prioridad. El código queda muy simple y tienen grandes resultados [WOJE95].

Esta es la arquitectura de agente que sirve de base para los experimentos de esta investigación, pues no se basan los agentes en un sistema de razonamiento simbólico formal, pues las redes neuronales se entrenan para que dadas las diversas circunstancias que se presenten, se ejecuten diversas acciones, las prioridades, entre las acciones, se definen mediante los casos proveídos en el conjunto de entrenamiento de las redes.

Algunos sistemas bajo esta aproximación son:

PENGI

Sistema creado por Chapman y Agre en el cual se basan en la idea de que la mayoría de las decisiones se convierten en rutina, por lo cual se codifican en una estructura para ser ejecutadas [WOJE95].

Situated Automata

Rosenschein y Kaelbling crearon un paradigma de autómatas, codificados en una máquina digital. Los agentes se especifican en términos de su percepción y acción. Cada agente toma una especificación de la semántica de sus entradas, una lista de hechos y un conjunto de transiciones; el programador especifica la semántica deseada para la salida (unión de hechos y la salida deseada) [WOJE95].

Esta aproximación ha tenido problemas teóricos pero parece combinar bien elementos simbólicos y reactivos de los sistemas declarativos [WOJE95].

7.1.3.3 Arquitecturas Híbridas

Las arquitecturas híbridas pretenden mezclar las virtudes de las deliberativas y las reactivas, por ejemplo se puede construir un agente deliberativo, con un modelo simbólico del mundo, encargado de desarrollar planes de acuerdo con su mundo; y un agente reactivo que responda a eventos que sucedan en el entorno sin preocuparse de razonamientos simbólicos. Usualmente este último es el que tiene prioridad para poder responder eficientemente a los eventos del entorno [WOJE95].

Este modelo propone una arquitectura en capas, donde haya una capa de bajo nivel que enlace sensores de datos con efectores (efectors) y una de alto nivel que se encargue de las metas a largo plazo. El problema con el que se enfrenta una arquitectura como ésta es el de definir un marco de trabajo que

permita unir los diversos agentes y manejar las interacciones entre las capas [WOJE95]. Esta es una alternativa que se puede valorar para un trabajo futuro, donde el agente de Robocopa combine los elementos reactivos, que se definen en esta investigación, mediante las redes neuronales, con la identificación y ejecución de jugadas planificadas. En este caso, el problema del marco de trabajo ya estaría resuelto para la comunicación de los agentes por el marco proveído por la Robocopa.

Algunos ejemplos de implementaciones de esta arquitectura son:

PRS

El *Procedural Reasoning System* es un sistema con arquitectura basada en deseos, creencias e intenciones desarrollado por Georgeff y Lansky. Este sistema contenía una biblioteca de planes con un conjunto de planes parcialmente elaborados, conocido como *Knowledge Areas* (KAs) cada uno tiene una condición de invocación [WOJE95]. Esta es otra alternativa que se puede utilizar a futuro para identificar jugadas parcialmente planificadas.

TOURINGMACHINES

Es un sistema desarrollado por Ferguson que consiste de dos subsistemas: uno de percepción y otro de reacción [WOJE95].

Está compuesto por tres capas [WOJE95]:

- Capa reactiva, que genera posibles cursos de acción como respuesta a los eventos que se den. Implementada con conjuntos de reglas situación-

acción.

- Capa de planeación, construye y selecciona acciones a ejecutar para alcanzar las metas. Utiliza una biblioteca de planes parcialmente elaborados.
- Capa de modelado, representación simbólica del estado cognitivo de otras entidades en el entorno.

Wooldridge y Jennings proveen otros ejemplos en [WOJE95].

Esta es una alternativa que también es viable de explorar buscando implementar jugadores de Robocopa que utilicen diferentes capas para reaccionar a diferentes situaciones, los planes parcialmente elaborados permitirían modelar estrategias a utilizar dadas ciertas circunstancias de juego.

7.2 Colaboración entre Agentes Inteligentes

La colaboración entre los agentes es un punto clave para lograr obtener un alto desempeño. Se pueden crear marcos generales para la colaboración entre agentes, o bien, marcos específicos para un problema determinado.

Las investigaciones realizadas por Zarout et al. [ZABO05] y Griffiths [GRIF05], se han orientado más a la elaboración de marcos de trabajo para agentes independientes que colaboran con otros de diferente naturaleza, de los cuales no tiene detalles de cómo realizan su labor ni de su arquitectura interna; así mismo, los otros agentes no conocen sus detalles. A este tipo de agentes se les llama **agentes heterogéneos**, los cuales se pueden comunicar mediante una interfaz definida que les permita saber que tareas puede hacer cada uno.

Para algunos problemas en los que se requiere que varios agentes de comportamiento similar colaboren entre sí, se pueden diseñar **agentes homogéneos**. El que los agentes sean homogéneos "se refiere a que todos están realizados de acuerdo al mismo paradigma. Sin embargo, dado que su localización puede ser distribuida y sus experiencias distintas por limitarse su visión a su entorno local, los agentes pueden diferenciarse en conocimiento y comportamiento" [GIRÁ99].

José Ignacio Giráldez Betrón [GIRÁ99] muestra algunos ejemplos como el modelo LOPE de García-Martínes and Borrajo, donde existe un mecanismo para compartir de lo aprendido entre los agentes, de manera que uno se pueda

beneficiar de lo aprendido por otro; el sistema MMS de Brauer and Weiss, los agentes son estructuralmente iguales, pero “sus objetivos locales, conocimiento y comportamiento no lo son”[GIRÁ99]; además mencionan otro ejemplo donde los agentes “además de ser homogéneos tienen el mismo comportamiento ante idénticas circunstancias” [GIRÁ99].

En esta investigación se parte del hecho de que los agentes a implementar serán homogéneos por lo que la siguiente sección se orienta bajo esa premisa.

7.2.1 Colaboración entre Agentes

Zarour et al. [ZABO05] y Griffiths [GRIF05] han desarrollado importantes investigaciones en torno a la colaboración entre agentes.

Zarour et al. [ZABO05] identifican dos tipos de agentes: el líder y el miembro. El primero es quien tiene el compromiso de realizar la tarea global y el segundo es aquel que acepta cooperar. Es importante para ambos tipos de agentes que puedan reaccionar a los cambios en el entorno y replantear su estrategia de ser necesario [ZABO05].

El esquema de [ZABO05] está diseñado para tareas en las que los agentes van a efectuar una labor compleja, estos agentes pueden participar o no de las tareas dependiendo de su nivel de motivación hacia las mismas, una vez que se ha negociado la colaboración se crean obligaciones hacia los miembros sobre las cuales serán evaluados luego [ZABO05].

Dentro de este esquema se vuelve un problema importante resolver cómo

es que un agente escoge un conjunto de acciones a realizar y cómo es que los los agentes ejecutan acciones coordinadas [ZABO05].

El esquema básico de [ZABO05] establece una fase de elaboración del plan, realizada por el líder, donde se define el plan general a ejecutar. La siguiente fase es la de negociación en donde el líder pacta con los miembros potenciales para la realización de las subtareas; una vez recibidas las respuestas de los posibles miembros, el líder evalúa las ofertas y decide si es factible el plan y a quiénes incluir (fase de evaluación); luego se procede a la ejecución del plan, al final de la cual si la meta no se logró, se toman las medidas correctivas necesarias (hacia los agentes) y se vuelve a la etapa de elaboración.

El marco de trabajo SoCCoF presentado por Zarour et al. en [ZABO05] es un esquema jerárquico, en el cual un agente miembro puede pedir la ayuda de otros, de los cuales el primero será el líder. Se crea una obligación para un agente cuando se le delega una tarea. Utiliza como elementos importantes en la evaluación el historial del potencial miembro, el mantenimiento de obligaciones, restricciones y operaciones de creación, asignación, delegación y cancelación, se hace en su implementación por medio de XML.

Una debilidad que tiene esquemas como el anterior, es que el tiempo que puede tomar la etapa de negociación, causado por la necesidad de comunicarse con muchos agentes, puede ser excesivo [GRIF05], lo cual lleva al sistema a ser poco escalable.

En esquemas como el de Zarout et al. [ZABO05], mucha de la

comunicación entre el líder y los posibles miembros, termina convirtiéndose en puro “overhead” pues al final no se crea una obligación, algunas veces porque el miembro no es un buen candidato o porque no tiene la suficiente motivación, pues para que el agente sienta inclinación a colaborar debe tener la motivación adecuada hacia lo que se le solicita; así mismo confianza con respecto al riesgo que representa colaborar [GRIF05].

Una aproximación para reducir el tiempo de negociación es utilizar un esquema de congregaciones o clanes donde los agentes se unen de acuerdo con intereses comunes. [GRIF05].

Griffiths [GRIF05] se basa en su modelo de cooperación en los factores de confianza, motivación y reputación. La confianza de un agente en otro depende de su propia experiencia con estos agentes y su propia disposición; la reputación es determinada por la confianza del propio agente y la información que otros agentes dan basados en su experiencia pasada. Esto significa que cuando un agente va a analizar si coopera con otro, utiliza su propia información y pregunta a otros sobre el agente, así decide si cooperar o no [GRIF05].

En el proceso de colaboración se dan tres etapas que Griffiths[GRIF05] define como:

1. **Selección del plan**: es el proceso de generación de metas y decisión de si solicitar ayuda o no [GRIF05].
2. **Adopción de intención**: es el proceso en el que el agente solicita

cooperación y recibe la respuesta de los otros agentes, se forma la intención de cooperación si suficientes agentes aceptan colaborar [GRIF05]. Esta fase es análoga a la de negociación propuesta por Zarour et al. en [ZABO05].

3. **Acción de grupo**: ejecución del plan. Una vez terminado se disuelve la cooperación. Dependiendo del resultando se actualizan los valores de confianza de un agente con respecto a otro [GRIF05].

Griffiths [GRIF05] no define una representación específica para los valores de motivación, confianza y reputación de los agentes, pues puntualiza que la interpretación que cada agente dé a los valores es totalmente subjetiva, de manera que un agente puede interpretar un valor numérico específico de diferente manera que otro agente; se recomienda entonces, definir cuál representación es más acorde al problema y buscar crear agentes que den una interpretación consistente a dichos valores.

Como se mencionó anteriormente, esta investigación se basa en agentes homogéneos donde la colaboración se da como resultado de las acciones individuales de los agentes, los cuales tienen conocimiento de cuáles son los jugadores que le pueden colaborar (los de su mismo equipo), debido a esto no es necesario crear un esquema específico para la colaboración; sin embargo, si en el futuro se desea introducir la figura de un entrenador (coach), se puede hacer que cada jugador almacene resultados de su desempeño, los cuales el entrenador puede utilizar para el reclutamiento de jugadores para cada partido.

7.3 Robocopa

En esta sección se abordan aspectos clave en la comprensión del ambiente de simulación Robocopa necesarios para la elaboración de la presente investigación.

La Robocopa es un proyecto conjunto internacional para promover el desarrollo de las técnicas de inteligencia artificial (IA), robótica y áreas afines; es un intento para fomentar la investigación en IA y robots inteligentes proveyendo un problema estándar donde muchas técnicas de IA pueden ser integradas [CHFO03].

"Las tecnologías involucradas en Rocobopa, cubren un amplio espectro de investigación, tales como principios de diseño de Agentes Inteligentes, colaboración multi-robot, adquisición de estrategias, razonamiento en tiempo real, inteligencia robótica y fusión de sensores"[4] [KAHU02]. De esta manera la Robocopa lanza muchos retos en los cuales se pueden utilizar muy diversas técnicas [KIAS97].

El problema específico que aborda la Robocopa es el problema de jugar fútbol, para el cual se crean agentes inteligentes que actúan como jugadores algunos, otros pueden actuar como director técnico.

En general el proyecto Robocopa incluye diversas áreas, entre ellas la electrónica y electromecánica, pues hay ligas en que se construyen robots que

4 Traducción libre del autor al texto: "The technologies that are involved in RoboCup cover a wide spectrum of research such as design principles of autonomous agents, multi-robot collaboration, strategy acquisition, real-time reasoning and planning, intelligent robotics, sensorfusion"[KAHU02].

compiten físicamente en un campeonato mundial.

La meta final de la robocopa es desarrollar para más tardar en el año 2050, un equipo totalmente autónomo de robots humanoides que le puedan ganar al campeón del mundo [CHFO03].

Existen diversos tipos de ligas en Robocopa, el sitio oficial [ROBO13] menciona las siguientes:

- Humanoides: son robots autónomos con un cuerpo similar al humano.
- Middle-Size: es para equipos de 6 robots que no tengan más de 50cm de diámetro.
- Small-Size: enfocado en colaboración de robots, también conocida como la liga F180.
- Standard: todos los equipos usan los mismos robots y se concentran en el desarrollo del software.
- Simuladores: se juega en un campo virtual y el enfoque es la inteligencia artificial y estrategia de equipo.

La liga de simuladores de Robocopa, en la cual se centra esta investigación, está basada en un componente principal: el “Soccer Server” (servidor) que es el simulador del sistema físico. Otros componentes son el “soccer monitor”, monitor para visualizar los juegos, y los “soccer players”, agentes jugadores que interactúan con el servidor. La ventaja del proyecto del simulador es que no es necesario preocuparse por aspectos físicos como reconocimiento de

objetos, comunicación, problemas de hardware, etc. Los investigadores se pueden concentrar es aspectos como la colaboración y el aprendizaje. [CHFO03]

Los jugadores son robots que obtienen información del entorno por medio de mensajes enviados por el servidor, en los cuales se muestra una sección del entorno de acuerdo con la visibilidad del robot; de esta manera, puede ver algunas banderas que delimitan el terreno de juego, el balón y los contrincantes.

El servidor de Robocopa posee un modelo sensorial para objetos que se pueden visualizar en el campo de juego y también para lo que se puede escuchar [CHFO03].

hear: permite escuchar mensajes enviados en el medio por el árbitro, el “coach” u otro jugador [CHFO03].

see: capta información sobre los objetos que están en el campo de visión del jugador [CHFO03].

Mediante el modelo **see**, un jugador puede ver algunos indicadores de su posición en el área de juego, el balón y otros jugadores.

En la Ilustración 7.1, se muestra la representación del campo de juego mediante banderas que ayudan a los jugadores a determinar su posición en el campo de juego.

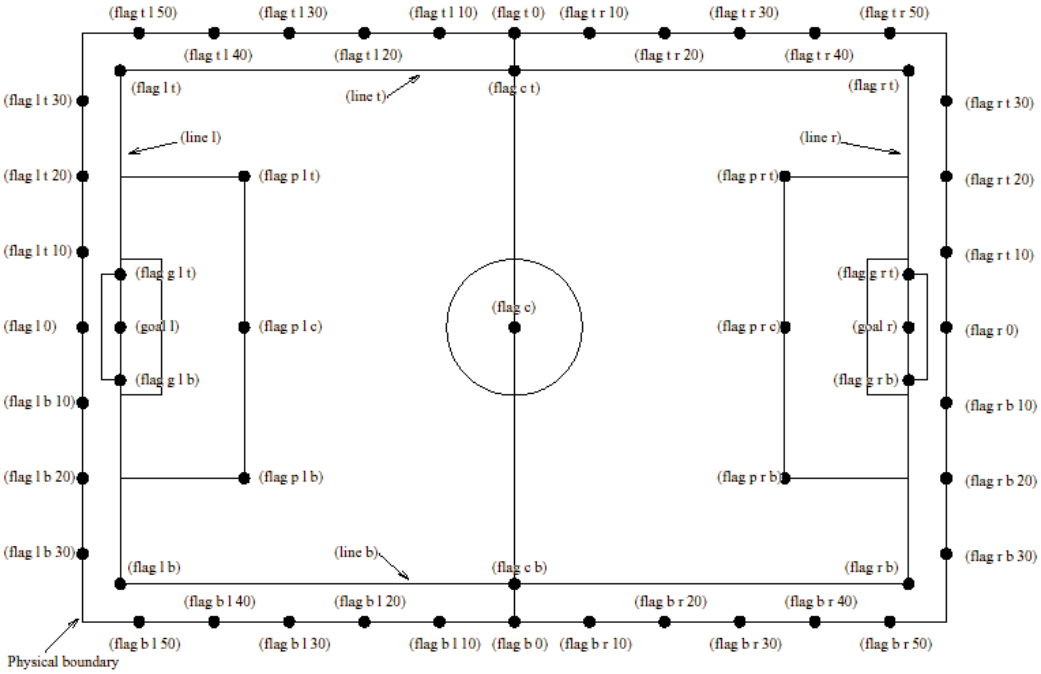


Ilustración 7.1: Campo de Juego [CHFO03]

Un jugador puede además enviar un mensaje sonoro a los otros haciendo uso del comando **say**, con el formato (say Mensaje). De esta manera un mensaje enviado por el jugador puede ser escuchado por todos los demás que estén a una distancia audible que se configura en el servidor [CHFO03].

Un jugador se inscribe como parte de un equipo para un juego enviando el comando de inicialización *(**init** Equipo)* al servidor [CHFO03].

7.3.1 Servidor

El servidor provee un campo virtual y simula todos los movimientos de la bola y los jugadores. Cada cliente controla los movimientos de un jugador. La comunicación entre el cliente y el servidor se realiza por medio de UDP/IP sockets.

El servidor controla el juego de acuerdo con un conjunto de reglas que tiene definidas. [CHFO03]

El programa servidor debe recibir los mensajes de los clientes, manejar sus peticiones y actualizar el ambiente. Además, envía a los clientes información sensorial sea objetos en el campo o características del jugador (potencia, velocidad, etc.); es un sistema de tiempo real que funciona con ciclos de tiempo de una determinada duración. Las acciones que se tienen que ejecutar deben recibirse en el intervalo correcto. [CHFO03]

La Ilustración 7.2, muestra el diagrama de clases UML de los objetos envueltos en la simulación.

7.3.2 Monitor

El monitor es una herramienta de visualización que le permite a los usuarios ver lo que está sucediendo en el encuentro. El monitor muestra el marcador, nombres de los equipos, posiciones de los jugadores y el balón [CHFO03].

Hay dos versiones del monitor: el clásico “rcssmonitor_clasic” y el “rcssmonitor” que extiende al clásico en las siguientes funciones [CHFO03]:

- Es posible hacer ZOOM de áreas del campo.
- La posición y velocidad de cualquier jugador y el balón se pueden imprimir en pantalla en cualquier momento.
- Puede mostrar información diversa como: el cono de vista de un jugador, la

"stamina" del jugador, tipo de jugador, etc.

- Los jugadores y el balón pueden ser manipulados con el ratón.

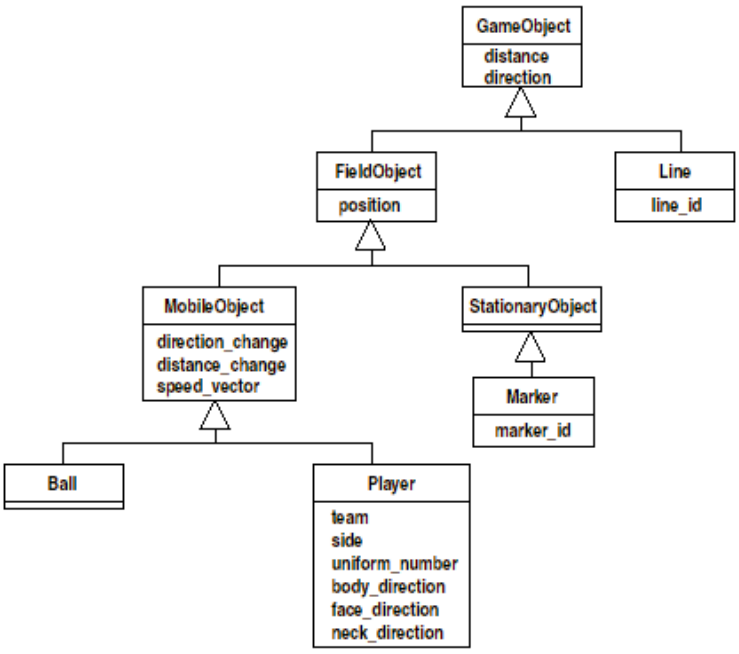


Ilustración 7.2 Diagrama de Objetos [CHFO03]

7.3.3 Cliente

Con el servidor se provee un programa llamado "rcssclient" que es una terminal base texto para la simulación. En esa terminal se pueden escribir comandos para enviarlos al servidor. La información recibida es mostrada en otra área de la pantalla [CHFO03].

Los programas que desean participar como jugadores en un partido, simplemente necesitan comunicarse con el servidor utilizando UDP/IP sockets,

esto les permite recibir la información del ambiente administrado por el servidor, generar acciones con base en esta información y enviarlas de nuevo al servidor. Eso sí, es importante hacerlo con eficiencia.

La tabla de la Ilustración 7.3, muestra las acciones del jugador que puede controlar el programa cliente. Se muestra que las acciones de capturar balón, cambiar vista, correr, patear, moverse a una determinada posición (solamente cuando la pelota no está en juego), enviar un mensaje, capturar propiedades físicas del jugador, pedir marcador, girar, girar la cabeza, están disponibles para que cualquier jugador las realice. La tabla muestra además los parámetros necesarios y si se pueden ejecutar una o varias veces por ciclo.

7.3.4 Árbitro

El árbitro automatizado envía mensajes a los jugadores para que sepan el modo de juego actual (tiro libre, movimiento inicial, tiro de esquina, jugando, etc). Además anuncia eventos como “gol” y “falta”. [CHFO03]

El programa de árbitro puede realizar acciones como: mover a un jugador a su lado cuando un gol se anotó y el jugador quedó en la mitad del equipo contrario, anunciar el gol, actualizar el marcador, poner la bola en el centro, poner el balón donde corresponda (tiros libres, saques de banda, tiros de esquina, etc.), apartar a los jugadores que no respetan la distancia para colocación de la barrera en un tiro libre, etc. [CHFO03]

From client to server	Only once per cycle
(catch *Direction*) *Direction* ::= ***minmoment*** ~ ***maxmoment*** degrees	Yes
(change_view *Width Quality*) *Width* ::= narrow | normal | wide *Quality* ::= high | low	No
(dash *Power*) *Power* ::= ***minpower*** ~ ***maxpower*** Note: backward dash consumes double stamina.	Yes
(kick *Power Direction*) *Power* ::= ***minpower*** ~ ***maxpower*** *Direction* ::= ***minmoment*** ~ ***maxmoment*** degrees	Yes
(move *X Y*) *X* ::= **-52.5** ~ **52.5** *Y* ::= **-34** ~ **34**	Yes
(say *Message*) *Message* ::= a message	No
(sense_body) The server returns (sense_body *Time* (view_mode {high | low} {narrow | normal | wide}) (stamina *Stamina Effort*) (speed *AmountOfSpeed DirectionOfSpeed*) (head_angle *HeadAngle*) (kick *KickCount*) (dash *DashCount*) (turn *TurnCount*) (say *SayCount*) (turn_neck *TurnNeckCount*) (catch *CatchCount*) (move *MoveCount*) (change_view *ChangeViewCount*))	No
(score) The server returns (score *Time OurScore TheirScore*)	No
(turn *Moment*) *Moment* ::= ***minmoment*** ~ ***maxmoment*** degrees	Yes
(turn_neck *Angle*) *Angle* ::= ***minneckmoment*** ~ ***maxneckmoment*** degrees turn_neck is relative to the direction of the body. Can be invoked in the same cycle as a turn, dash or kick.	Yes

Ilustración 7.3 Acciones del jugador [CHFO03]

7.4 Redes Neuronales

En esta sección se incluyen los principales conceptos teóricos sobre redes neuronales, mostrando principalmente las que interesan en esta investigación, las redes neuronales de retropropagación, pero también mostrando algunos tipos diferentes que se podrían utilizar en trabajo futuro.

Las redes neuronales son modelos matemáticos inspirados en el funcionamiento del cerebro humano, que tienen la característica que aprenden a asociar patrones sin necesidad de ser directamente programadas. [RUWI94]

Una red neuronal debe funcionar, de manera general, similar a las neuronas humanas. De este modo se crean neuronas y enlaces (sinapsis) entre ellas [RUWI94],[GORN97]. Se busca que un gran número de unidades de proceso simples actúen simultáneamente en un patrón distribuido [TODD91].

Cada neurona tiene una función de activación que determina cuando la neurona emitirá una señal (valor) de salida y cuando no. Así mismo, la combinación de las neuronas forma una red en la que la salida de unas alimenta a otras en sus entradas [RUWI94].

La arquitectura usual consta de una serie de neuronas de entrada, conectadas a las de salida, por medio de un conjunto de neuronas intermedias o escondidas, aunque en realidad pueden existir varias capas escondidas [RUWI94] [HINT06] [HINT07]. Todas las neuronas de la red hacen exactamente lo mismo, suman sus valores de entrada – provenientes de otras unidades – y aplican una

función a esa suma que resulta en una salida con un rango restringido [TODD91].

Las señales de entrada que alimentan a cada neurona se calculan de la siguiente manera: la salida de las neuronas de la capa previa (es decir, sus valores de activación X_1,X_2 y X_3) son multiplicadas por sus respectivos pesos de las sinapsis P_1, P_2 y P_3; esos resultados son sumados, resultando el valor *u*. El valor de activación de la neurona es calculado por una función, dando como resultado el valor de salida *v*. Esta función es usualmente no lineal y debe ser elegida con cuidado pues el desempeño de la neurona depende fuertemente de ella [GORN97]. La Ilustración 7.4, muestra de manera gráfica lo descrito anteriormente.

El aprendizaje se produce por medio de conjuntos de entrenamiento los cuales son evaluados para que luego los pesos (valores) de las conexiones (sinapsis) entre una neurona y otra se ajusten para producir la salida esperada para cada uno de los elementos del conjunto de entrenamiento [RUWI94]. Los pesos son ajustados por medio de un proceso conocido como “descenso en gradiente” donde se hacen pequeños ajustes a los pesos para minimizar el error, el cual dice que tan largo está la salida actual de la deseada [TODD91].

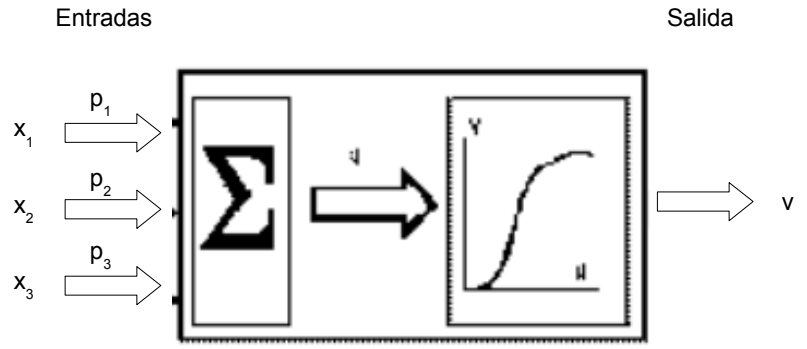


Ilustración 7.4 Representación de neurona.

Existen diversas maneras en las que una red puede aprender, están las de aprendizaje supervisado, donde se entrena con pares de entrada y salida esperada; las de aprendizaje no supervisado, son las que la red no recibe una retroalimentación de si la salida es correcta o no, usualmente utilizadas para agrupar los patrones de entrada en diversas categorías; existen las de aprendizaje híbrido, que mezclan el supervisado en unos pesos y lo no supervisado para otros [JAMA96]. En esta investigación interesan las de retropropagación que son de aprendizaje supervisado.

7.4.1 Redes Neuronales de Retropropagación

Es el método más simple para entrenar redes neuronales [RUWI94].

Ottair et al. [OTSA05] exponen que:

> La retropropagación BP aprende un conjunto predefinido de ejemplos de pares de salidas mediante el uso de un método de dos fases de propagación... luego de que un patrón ha sido aplicado como estímulo a la

primera capa de unidades de la red, es propagado a la capa siguiente, proceso que se repite hasta llegar a la última capa y así una salida es generada. Esta salida es luego comparada con la salida deseada, y una señal de error es calculada para cada unidad de salida. Luego, las señales se transmiten hacia atrás desde la capa de salida a cada unidad de la capa intermedia que contribuye directamente a la salida. Sin embargo, cada unidad en la capa intermedia recibe una porción de la señal total de error, basado en la contribución relativa que la unidad hizo para la salida original. Esto se repite, capa por capa, hasta que todas las neuronas hayan recibido una señal relativa a su participación para generar la salida.[5]

La Ilustración 7.5, muestra de una manera simplificada este modelo.

Durante cada paso del entrenamiento se hacen pequeños ajustes a los pesos, luego el ejemplo es evaluado de nuevo; luego de una serie de intentos (para todos los patrones de entrenamiento) la salida de la red será lo suficientemente cercana a la esperada por lo que el entrenamiento puede finalizar [TODD91].

El proceso de aprendizaje se puede ver tan simple como encontrar el conjunto de conexiones adecuadas para que se pueda producir la salida esperada

5 Traducción libre del autor para el texto: "The Backpropagation BP learns a predefined set of output example pairs by using a two–phase propagate adapts cycle... after an input pattern has been applied as a stimulus to first layer of network units, it is propagated through each upper layer until an output is generated. This output pattern is then compared to the desired output, and an error signal is computed for each output unit. The signals are then transmitted backward from the output layer to each unit in the intermediate layer that contributes directly to the output. However, each unit in the intermediate layer receives only a portion of the total error signal, based roughly on the relative contribution the unit made to the original output. This process repeats, layer by layer, until each unit in the network has received an error signal that describes its relative contribution to the total error."

para la entrada recibida. Rumelhart et al. [RUWI94] no recomiendan utilizar todo el conjunto de prueba para entrenar la red, para que ésta no se limite a memorizar los resultados, por el contrario, es bueno utilizar un 80% para entrenar y un 20% para probar el desempeño [GORN97].

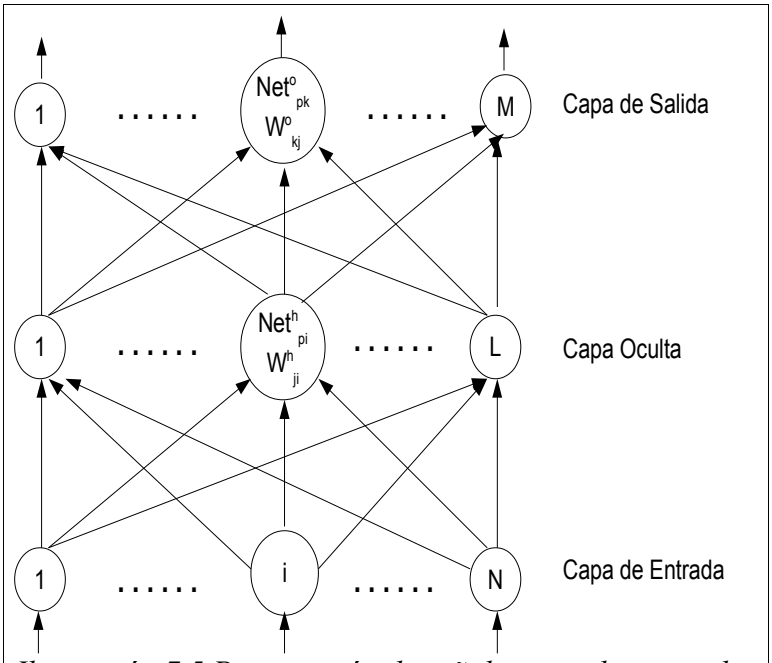


Ilustración 7.5 Propagación de señales en red neuronal de retro-propagación.

La Ilustración 7.4, presenta una vista esquemática de una neurona. Se puede ver cómo la sumatoria de las entradas es aplicada a una función para obtener el valor de salida [TODD91].

Una característica importante señalada por Peter Todd [TODD91] es que debido al gran paralelismo que se da en el modelo, pequeños cambios en la red no afectan mucho su desempeño, incluso el remover una unidad puede no afectar mucho el error de predicción de la red porque las otras unidades pueden subsanar

el vacío de la eliminada.

Algunas desventajas de las redes de retropropagación son:

- usualmente toman mucho tiempo para converger (alcanzar los resultados esperados).
- una vez entrenadas no pueden incorporar nuevo conocimiento [TODD91], [CAGR91].

Aún con lo anterior, han sido altamente utilizadas en problemas de reconocimiento de patrones [RUSS05] y recientemente para hacer “aprendizaje profundo” en procesos de reconocimiento de caracteres [HINT06][HINT07].

7.4.1.1 Función de Activación

Una definición simple de la función de activación la brindan Cazorla et al. donde exponen que “La función de activación o función de filtro determina si la neurona se activa o no, y qué valor va a producir” [CAZO99].

El estado o valor de cada neurona en la red está dado por una función de activación. En el caso más simple es la sumatoria de los productos de la entrada por la salida [RUSS05], lo cual se conoce como función de combinación [CAZO99].

Sea X = (x1,x2,...xn) los valores de entrada, y W = (w1,w2,...wn) los pesos de las conexiones por las cuales se recibe la entrada, el resultado U = $\sum w_i x_i$ es el valor base para la función de activación [RUSS05].

Otra forma de función de activación es la llamada *función de escalón* [CAZO99], donde un valor umbral determina si la función retorna 1 si U es mayor que el umbral o 0 si es menor o igual [RUSS05].

La siguiente figura muestra como se ve gráficamente la función de escalón*:*

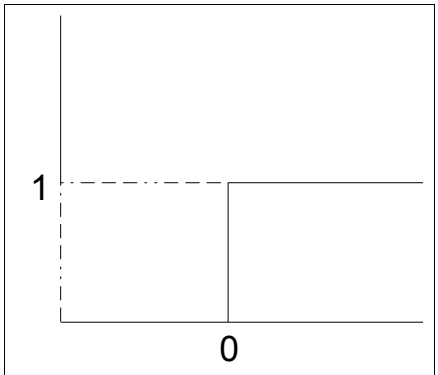


Ilustración 7.6 función de escalón.

Es importante que la función de activación no sea lineal, pues de ser así, una red que tuviera múltiples capas (de entrada, intermedias y de salida) se comportaría igual que una de una sola capa, que han probado ser muy limitadas [RUSS05]. De la explicación del algoritmo de aprendizaje de retropropagación dada por Cazorla et al. [CAZO99], que presenta claramente como el algoritmo está basado en la propagación hacia atrás de los deltas entre lo esperado y lo obtenido, queda claro que es necesario que la función de activación debe ser diferenciable [CAZO99]. Se recomienda que la función y su derivada sean fáciles de computar, que tenga una alta parte lineal [LLHO97].

Además de la función de escalón hay otras utilizadas, la más común es la

función sigmoide, que tiene una forma como:

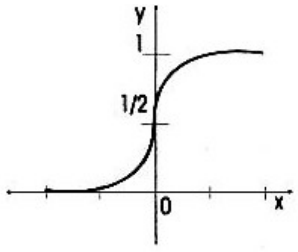


Ilustración 7.7 función sigmoide.

La función está definida por [CAZO99]:

$$f(x)=\frac{1}{1+e^{-x}}$$

Se puede ver que es continua y diferenciable en todos los puntos, lo cual es necesario para la derivación del algoritmo [CAZO99].

Otras funciones muy utilizadas son la Gausiana, que es la misma distribución normal, dada por $f(x)=e^{-x^2}$ o la complemento Gausiana, dada por $f(x)=1-e^{-x^2}$ [GUTI04], otra es la Tangente Hiperbólica dada por $tansig(n)=\frac{2}{(1+e^{-2n})-1}$ [LLHO97].

7.4.1.2 Algoritmo de Aprendizaje

A continuación se presenta de manera simplificada el algoritmo de aprendizaje de este tipo de redes neuronales de acuerdo con Cazorla et all.

[CAZO99]:

El objetivo es encontrar los pesos que minimicen el error producido por las neuronas. Consta de dos fases:

1. Propagación hacia adelante: calcular las salidas dadas las entradas.
2. Propagación hacia atrás (retropropagación): se va calculando el error por cada neurona y se actualizan los pesos desde la última capa a la primera.

Sea una red multicapa de K capas como la de la siguiente figura:

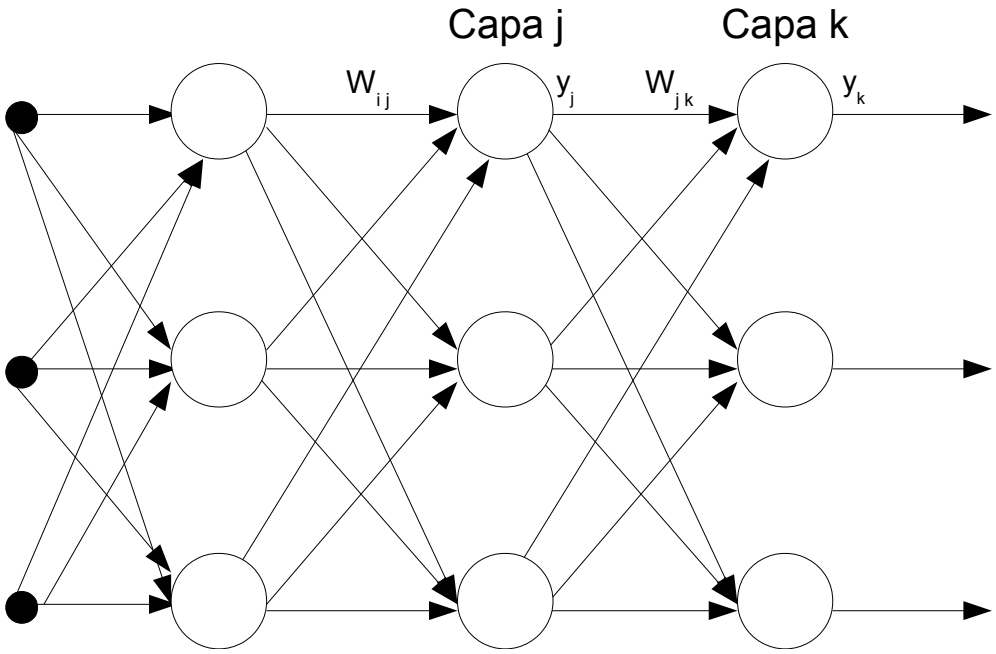


Ilustración 7.8: Propagación hacia adelante [CAZO99].

El algoritmo busca ver cuál es el error del que cada neurona es responsable, de acuerdo con ese error se modifican los pesos de entrada a dicha neurona.

Cada neurona es responsable por un error denominador δ_k dado por:

$$\delta_k = (d_k - y_k)\, f'(x_k)$$

Donde d_k es lo deseado y y_k es lo obtenido, $f'(x_k)$ es la derivada de la función de activación.

Los pesos se actualizarán basados en el error mediante:

$$w_{jk} = w_{jk} + \eta \delta_k y_j$$

w_{jk} es un peso cualquiera de la capa j a la capa k.

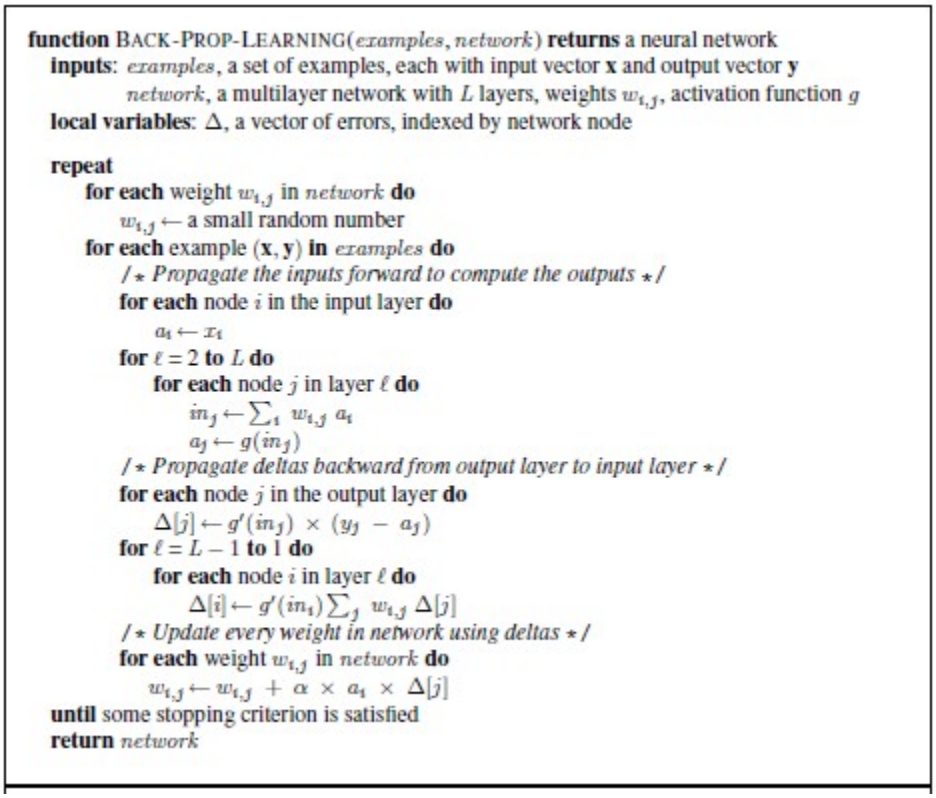
η es una constante de proporcionalidad.

Para actualizar los errores en la capa anterior, hay que saber cuáles son los errores (δ) correspondientes, que se calculan mediante:

$$\delta_j = f'(x_k) \sum_k w_{jk} \delta_k$$

“La idea es que una neurona oculta de la capa j (supongamos para simplificar que es la inmediatamente anterior a la k), es responsable de parte del error δ_k de cada una de las neuronas a las que se conecta su salida. Por ello, el error cometido en la neurona de la capa j será la suma de los errores de las neuronas de la capa siguiente ponderados por los pesos que conectan con ellas” [CAZO99].

Russel y Norvig proveen en el AIMA [RUNO10] el siguiente esquema con el pseudo-código del algoritmo de aprendizaje, el cual se muestra en el lenguaje original.



function BACK-PROP-LEARNING($examples, network$) **returns** a neural network
inputs: $examples$, a set of examples, each with input vector **x** and output vector **y**
$network$, a multilayer network with L layers, weights $w_{i,j}$, activation function g
local variables: Δ, a vector of errors, indexed by network node

repeat
for each weight $w_{i,j}$ in $network$ **do**
$w_{i,j} \leftarrow$ a small random number
for each example (**x**, **y**) **in** $examples$ **do**
/* *Propagate the inputs forward to compute the outputs* */
for each node i in the input layer **do**
$a_i \leftarrow x_i$
for $\ell = 2$ **to** L **do**
for each node j in layer ℓ **do**
$in_j \leftarrow \sum_i w_{i,j}\, a_i$
$a_j \leftarrow g(in_j)$
/* *Propagate deltas backward from output layer to input layer* */
for each node j in the output layer **do**
$\Delta[j] \leftarrow g'(in_j) \times (y_j - a_j)$
for $\ell = L - 1$ **to** 1 **do**
for each node i in layer ℓ **do**
$\Delta[i] \leftarrow g'(in_i) \sum_j w_{i,j}\, \Delta[j]$
/* *Update every weight in network using deltas* */
for each weight $w_{i,j}$ in $network$ **do**
$w_{i,j} \leftarrow w_{i,j} + \alpha \times a_i \times \Delta[j]$
until some stopping criterion is satisfied
return $network$

Ilustración 7.9: Algoritmo de Retropropagación [RUNO10]

7.4.1.3 Algoritmo de Predicción

Una vez que la red se ha entrenado, el predecir la salida para un vector de entrada es simple. Rojas [ROJA96] lo define como un grafo, donde los arcos transportan información numérica de nodo a nodo, creando una cadena donde se transforma el vector de entrada en uno de salida.

7.4.1.4 Técnicas de Aprendizaje

Para lograr un correcto aprendizaje de la red neuronal, hay dos aspectos importantes que se deben tener en cuenta:

- Debe haber un balance adecuado entre el tamaño del conjunto de entrenamiento, cuántos ejemplos se utilizan para el entrenamiento y cuántos para probar el desempeño de la red. Gorni [GORN97] lo propone de una manera muy simple y que en esta investigación funciona muy bien, y es elegir alrededor de un 80% del conjunto de entrenamiento para entrenar. Luego se puede elegir el restante 20% para probar el desempeño.
- Russel y Norvig [RUNO10] presentan un punto importante, la estructura de la red también es relevante, pese a que el diseño de cuantas unidades por capa es un proceso de prueba y error, se debe cuidar que no sea tan grande la red como para que se limite a memorizar los datos de entrenamiento sin poder generalizar.

7.4.1.5 Avances y Modificaciones en Redes Neuronales

Recientemente Hinton se ha mantenido publicando importantes investigaciones en las redes neuronales, especialmente en el campo de la retro-propagación en redes de múltiples capas, lo que se conoce como aprendizaje profundo o "deep-learning".

Hinton en [HINT07] muestra una interesante alternativa para entrenar redes eficientes en reconocimiento de objetos, reconocimiento de escritura o reconocimiento de voz, utilizando redes neuronales multi-capa, las cuales se entrenan en dos direcciones, de arriba-abajo para generar "feature-detectors" y de

abajo-arriba para la clasificación.

Hinton y Salakhutdinov [HISA06] muestran también como mediante el uso de “autoencoders” creados a partir del encadenamiento de RBM (Restricted Boltzmann machine), que son en general feature-detectors, se puede reducir la dimensionalidad de datos de alta dimensión, haciéndolos más manejables, y pudiendo reconstruir los datos originales. El mecanismo básico es similar al explicado en el párrafo anterior.

Hof resume en [HOFR13] como google ha aprovechado las técnicas de aprendizaje profundo para mejorar búsquedas, reconocimiento de imágenes y el reconocimiento de voz incluido ahora en los teléfonos Android. Además provee algunas tendencias que guiarán el trabajo futuro en este área.

7.4.2 Otros tipos de Redes Neuronales

Existen muchas variedades de redes neuronales, las cuales han sido desarrolladas para responder a diversas necesidades, tales como manejar categorías difusas, permitir que la red continúe aprendiendo luego del entrenamiento inicial, etc.

A continuación, se hace una breve mención a otros tipos de redes neuronales, principalmente de aprendizaje supervisado, de los cuales no se brinda mayor detalle pues no son utilizados en el desarrollo de esta investigación, pero pueden ser útiles en un trabajo futuro.

7.4.2.1 Redes ART and ARTMAP

La teoría de resonancia adaptativa (ART) fue introducida por Grossberg entre 1976 y 1980, como una teoría sobre el procesamiento humano de información cognitiva. [CAGR91], [WEEN97]. Surgió como resultado de un intento por comprender cómo los sistemas biológicos son capaces de mantener elasticidad en el tiempo sin comprometer la estabilidad de los patrones aprendidos previamente. [WEEN97].

Las redes neuronales ART (Adaptive Resonance Theory) han sido utilizadas en una amplia gama de áreas para tener un aprendizaje y predicción rápido y estable. Así se han utilizado en aplicaciones como diseño y manufactura de aeroplanos, reconocimiento automático de objetivos, monitoreo de herramientas, diseño de circuitos digitales, análisis de químicos y visión de robots [CARP03]. El énfasis de las redes ART reside en el aprendizaje no supervisado y la auto-reorganización [WEEN97].

Aprendizaje no supervisado se refiere a que no se utiliza más información que la de los patrones de entrada. Hay otros modelos de aprendizaje como: supervisado, en el que un instructor brinda el patrón de entrada y la respuesta correcta; aprendizaje reforzado, donde a la red se le dice si clasificó bien la entrada o no [WEEN97].

Las redes ARTMAP son la versión de aprendizaje supervisado de las redes ART. Poseen mecanismos de control internos que crean categorías de reconocimiento de tamaño óptimo, maximizando la compresión de código y

minimizando el margen de error. [CAGR91] , [CARP03].

Todos los modelos basados en ART permiten aprendizaje incremental, rápido y estable, generalización múltiple y rápida convergencia con un número de patrones de entrenamiento relativamente bajo [PAGO02].

La característica central de todos los sistemas ART es un proceso de "pattern-matching" que compara la entrada actual con una clase aprendida producida por un código activo o hipótesis [CARP03].

7.4.2.2 Redes Neuronales Fuzzy ARTMAP

Un sistema Fuzzy ARTMAP consiste en dos módulos Fuzzy ART (aprendizaje no supervizado) unidos por una capa de neuronas llamada mapa interART. Cada módulo hace una clasificación no supervisada en los espacios de entrada y salida y el mapa almacena las relaciones entre los clusters creados por ambos módulos ART. El módulo A recibe el patrón a clasificar y el módulo B recibe la clasificación esperada [PAGO02].

7.4.2.3 Time delay neural networks (TDNN)

Las redes neuronales time-delay son aquellas que usan time-delays en las conexiones. Cada neurona toma en cuenta no solo la información actual de las neuronas de la capa anterior, sino que también cierta información del pasado, gracias a algunas interconexiones de retraso. El training se da en patrones espacio-temporales y las entradas son clasificadas en cada unidad de tiempo por la capa de salida. Después del entrenamiento, los pesos son incrementados en las

interconexiones en que los retrasos de tiempo sean importantes. [DLIN92].

7.4.2.4 Funciones de Base Radial

Son un modelo híbrido que utiliza aprendizaje supervisado y no supervisado. Tiene tres capas: entrada, intermedia y salida. [GARC05]

"Las neuronas de la capa oculta calculan la diferencia entre el vector de entradas y los pesos sinápticos, denominados, centroides, y a esta diferencia le aplican una función radial con forma gaussiana." [GARC05]

García [GARC05] muestra el siguiente diagrama de la red:

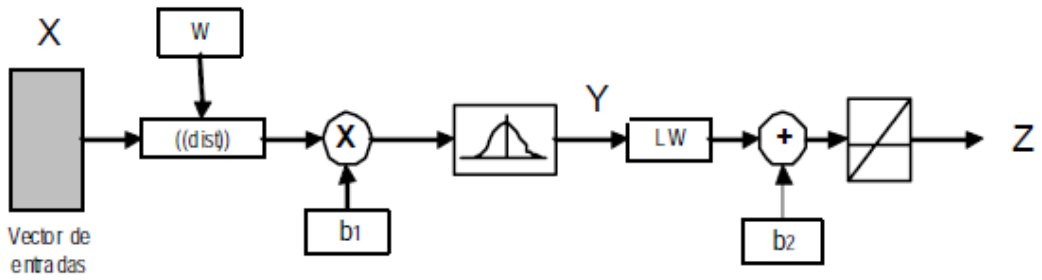


Ilustración 7.10: Función de Base Radial [GARC05]

7.4.2.5 Red Neuronal de Regresión General

García-Herrara y Mayol-Cuevas definen las redes de regresión general como una que “aproxima la función que existe entre las entradas (en este caso vectores de simetría) y las salidas (etiquetas binarias que indican la clase del estímulo), empleando la técnica estadística de regresión no-lineal”[GARMAY].

La principal característica de esta red es que el aprendizaje es en un solo paso y que su desempeño es mejor cuanto mayor sea el conjunto de entrenamiento [GARMAY].

7.4.2.6 Mapas autoasociativos (SOMs)

Los mapas autoasociativos (SOM por sus siglas en inglés) son un algoritmo de redes neuronales. La principal característica que tienen es que utilizan aprendizaje no supervisado, i.e. el resultado del entrenamiento depende únicamente de la entrada en las estructuras de datos. Es una variante de los algoritmos de clustering de vectores multidimensionales, como por ejemplo k-means. [STAR00].

Merelo J.J. los describe en su tutorial como:

> Un algoritmo, a veces agrupado dentro de las redes neuronales, que a partir de un proceso iterativo de comparación con un conjunto de datos y cambios para aproximarse a los mismos, crea un modelo de esos mismos datos que puede servir para agruparlos por criterios de similitud; adicionalmente, este agrupamiento se produce de forma que la proyección de estos datos sobre el mapa distribuya sus características de una forma gradual [MEME13].

Es importante que en los SOM todas las neuronas (nodos o centros de clase) están ordenados en una estructura (normalmente, un cuadriculado bidimensional). En el entrenamiento no solo la neurona ganadora es modificada

sino que todos sus vecinos también; sin embargo, la fuerza de la variación va a depender de la distancia a la neurona ganadora [STAR00].

Cada neurona representa un vector de columnas n-dimensionales w=[w1,w2,...,wn]T, donde n depende de la dimensionalidad inicial (la de los vectores de entrada). Las neuronas usualmente se localizan en los nodos de un cuadriculado bidimensional con celdas hexagonales o rectangulares. Las neuronas interactúan entre sí, esa interacción es determinada por la distancia a la que se encuentren [STAR00].

7.4.2.7 Redes de Elman

Son redes parcialmente recurrentes que como lo muestra [ELMA90] crean conexiones hacia atrás de una neurona a otra. Elman las propone como una alternativa para dar una representación del tiempo en una serie de datos, donde la conexión hacia atrás no es modificada por el algoritmo de retropropagación, sino que su peso es fijo. Una aproximación que propone Elman en [ELMA90] es crear un nodo interno, que se suma a la entrada y que recibe el valor de la capa oculta.

Los resultados mostrados por Elman [ELMA90] en diversos casos como estructura en secuencias de letras, xor, descubrir clases léxicas de acuerdo con el orden de las palabras, muestran su utilidad.

7.5 Redes Neuronales en Robocopa

El incorporar redes neuronales en la solución de situaciones en Robocopa ya ha sido abordado por varios autores; sin embargo su aproximación es muy diferente de la propuesta en esta investigación. A continuación se hace una breve mención a algunos de éstos, con el fin de mostrar aproximaciones exitosas.

Kalyviotis y Hu [KAHU02] utilizaron las redes neuronales para la toma de decisiones para la jugada de saque de puerta. Para ésto modelaron 10 diferentes jugadas, a las cuales se les asocia un riesgo. La red recibe como entrada los riesgos para cada jugada en el ciclo actual de juego y la cantidad de contrarios cerca del jugador que recibe el balón de último en cada jugada.

El siguiente gráfico muestra la topología de la red:

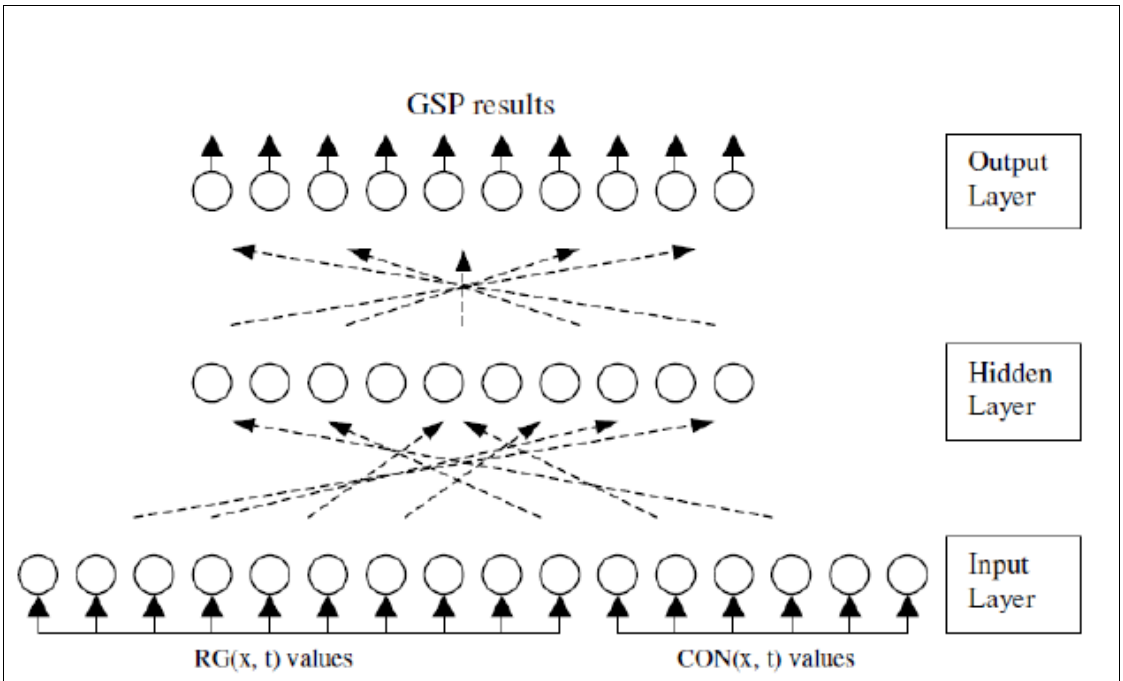


Ilustración 7.11: Topología de Red [KAHU02]

El desempeño de la red neuronal fue mejor que otras aproximaciones, obteniendo un 76% de éxito, con un conjunto de datos que en promedio tiene un 80% de éxito, contra la técnica de decisiones programadas manualmente (64%) y la toma de decisiones aleatoria (47%) [KAHU02].

Un ejemplo de una decisión más simple es la de David Abraham [ABRA01], donde utiliza una red neuronal de retropropagación para decidir si pasa la bola o remata en una situación de un defensor contra un atacante, situación que es similar, pero más simple que la propuesta en esta investigación para medio campistas y delanteros.

Ambas aproximaciones convierten el problema de una toma de decisiones para robocopa en un problema de clasificación, pues cada categoría es una jugada a elegir.

Hossein et al. [HOKA05] muestran un uso específico de las redes neuronales en Robocopa. En su caso abordan el problema del remate a marco. Así utilizan una red neuronal de retro-propagación para modelar el remate a marco junto con la dirección.

El modelo de [HOKA05] es de una red con 6 entradas que brindan distancia del balón y su ángulo con respecto al marco, posición del balón y posición del portero. Las salidas por su parte constan de un nodo que indica si remata o no, y 5 más que funcionan como dígitos binarios que forman un número que identifica el área del marco hacia el cual rematar. Para la capa intermedia de manera heurística se encontró que 5 neuronas daban un buen desempeño de la red. El

entrenamiento se hizo con retropropagación, con 5500 ejemplos y un error permitdo de 0,05. Al probar con 1500 casos de prueba se obtuvo un 77,46% de éxito contra un 61,53% que se había obtenido con un algoritmo de decisión manual.

7.6 Tácticas de Fútbol

En esta sección se recopilan algunas tácticas sugeridas por diversos organismos que promueven capacitaciones y entrenamiento en fútbol. Se omiten incluir definiciones básicas de qué es el fútbol, qué tipos de jugadores existen y las reglas del juego, pues es un deporte ampliamente conocido y no es el objetivo de la investigación ahondar en esos detalles.

Es importante recalcar que cuando se aborda una táctica, por ejemplo alguna defensiva, esto no quiere decir que únicamente atañe a los defensas, pues al ser el fútbol un deporte de conjunto, todo el equipo tendrá que aplicar su táctica defensiva; sin embargo se hará énfasis en los jugadores cuya función principal es aplicar dicha táctica.

La definición de Benarroch confirma lo anterior:

> Los principios tácticos defensivos son todas las acciones tácticas que realiza un equipo cuando la posesión del balón la tiene el equipo contrario, con el objetivo de recuperar el balón, retrasar o impedir el avance de jugadores rivales y proteger la portería propia para evitar encajar un gol... Los delanteros no deben dejar de jugar si pierden la posesión del balón,

convirtiéndose en los primeros defensores [BENA13].

7.6.1 Tácticas defensivas

Generalmente se habla de dos tipos de estrategia defensiva [BENA13] [GRAN07], la defensa en zona y la marca hombre a hombre, aunque también las hay mixtas. En la primera los defensas cubren un área específica, en la segunda se encargan de marcar un jugador específico [BENA13]. En los experimentos de esta investigación se prefiere la defensa en zona.

Los diferentes jugadores, pero en especial los de posiciones defensivas pueden ejecutar las siguientes acciones:

7.6.1.1 Coberturas

Esta es una técnica que consiste en estar cerca de un jugador compañero para poder "ayudarle en las tareas defensivas y ocupar sus funciones en caso de ser superado o desbordado por el poseedor del balón" [BENA13]. Para esto el jugador debe estar consciente de su entorno y estar sumamente concentrado, Benarroch [BENA13] menciona tres aspectos importantes:

- El jugador que va a hacer la cobertura debe estar más cerca de la portería que el compañero al que cubre.
- El jugador que va a hacer la cobertura debe mantener visión del compañero, del contrario, e interceptar pases y poder hacer una permuta.
- El jugador debe mantener su concetración sobre los movimientos de los

contrarios, y actuar rápida y decididamente.

7.6.1.2 Permutas

“Principio táctico defensivo que realiza un jugador recién desbordado, consistente en ocupar, lo más rápidamente posible, la posición y las funciones dejadas por el compañero defensor que le hacía la cobertura y que, en su ayuda, sale al encuentro del adversario” [BENA13].

La siguiente figura de Benarroch [BENA13] muestra un ejemplo de este movimiento:

Ilustración 7.12: permuta [BENA13]

En la Ilustración anterior se muestra cuando un defensa (3) es rebasado, el que hacía la cobertura (4) cambia a hacer la marca y el número 3 cambia a hacer cobertura de su compañero.

7.6.1.3 Marcaje

En el artículo “Táctica y Estrategia” se define de manera muy sencilla “todas

las acciones que hacen los jugadores del equipo sobre un rival, en pugna directa"[GIPU13]. Benarroch [BENA13] amplía que se puede tratar de buscar interceptar pases, macar de cerca, incluso "molestar" de alguna manera al rival para impedir que el atacante logre su objetivo.

Ilustración 7.13: Marcaje [BENA13]

Benarroch [BENA13] define tres tipos de marcajes: el individual, el por zonas, el mixto (en que sí es requerido un defensor y puede salir de la zona) y el combinado (donde algunos jugadores usan un tipo de marcaje y otros uno diferente).

El marcaje se puede realizar mediante diversas acciones [BENA13]:

- **Entrada**: tratar de quitar el balón al contrario, es la acción defensiva más importante.
- **Anticipación**: adelantarse al rival que va a recibir el balón, impidiendo así que lo reciba.
- **Corte o despeje**: hay dos formas de hacerlo, una es apoderarse del balón

interceptándolo, o bien, desviar el balón evitando que se cumpla el objetivo del rival.

- **Carga**: Usar el cuerpo de manera reglamentaria sea para dificultar manejo del balón al contrario, tomar el control del balón o bien protegerlo.

7.6.1.4 Vigilancia Defensiva

Benarroch [BENA13] y [GIPU13] mencionan una técnica conocida como vigilancia defensiva, la cual consta en "controlar los movimientos del rival" [GIPU13] aunque no se esté haciendo un marcaje como tal. La idea básica es "Vigilar las zonas no ocupadas para evitar que sean aprovechadas por el equipo adversario" [BENA13] y dicha vigilancia se puede realizar sobre un jugador específico [BENA13].

7.6.1.5 "Pressing"

Táctica colectiva defensiva que Benarroch [BENA13] define muy claramente, es una táctica bastante conocida y aplicada en el fútbol, por lo tanto se muestra la definición integral a continuación:

"Es la acción de asedio o acoso sobre el equipo contrario, que se realiza una vez perdida la posesión del balón, sobre uno, varios o la totalidad de los adversarios, con la finalidad de no dejarles ninguna libertad de acción, arrebatarles la posesión del balón, provocarles un error en su juego o romper en su origen su juego ofensivo" [BENA13].

Así mismo Benarroch [BENA13] define los siguientes tipos de pressing:

- Pressing sobre el poseedor del balón.
- Pressing sobre una línea.
- Pressing en zona.
- Pressing con fuera de juego.
- Pressing sobre todos los adversarios o pressing total.
- Pressing al perder el balón.
- Pressing desde que el adversario pone el balón en juego.

7.6.1.6 Repliegue

Táctica defensiva colectiva que Benarroch [BENA13] define como el conjunto de acciones defensivas en que los jugadores retornan a su zona de marcaje al perder el balón (o acción definida), buscando así estar organizados para recuperar el balón.

Los repliegues pueden ser individuales o colectivos [BENA13].

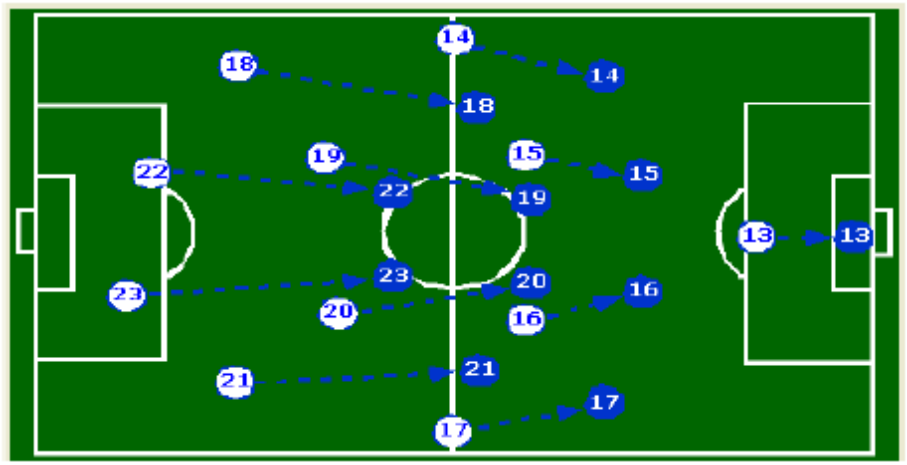


Ilustración 7.14: Repliegue [BENA13]

7.6.2 Tácticas Ofensivas

En el fútbol la meta principal es lograr anotar, de esta manera la táctica ofensiva es parte integral. Los equipos de fútbol requieren implementar estas tácticas tanto a nivel individual como grupal. Benarroch [BENA13] da un conjunto de técnicas que se muestran a continuación.

7.6.2.1 Vigilancia Ofensiva

Benarroch [BENA13] destaca que aún cuando se está atacando, no todos los jugadores se pueden volcar al ataque, desatendiendo su posición o misión defensiva. En concreto, los defensas y portero se deben mantener vigilantes de las posiciones de los contrarios.

7.6.2.2 Conservación del Balón

No siempre la acción ofensiva va directamente encarrilada a buscar la

anotación. En ocasiones las circunstancias del juego hacen necesario dar prioridad a controlar el balón.

Es un principio táctico ofensivo que realiza el equipo poseedor del balón realizando continuas acciones para mantener la posesión del balón, sin otro objetivo que conservarlo (conservación del balón) y continuar siendo el dueño del juego (control del juego).[BENA13]

La decisión que se tome cuando se gana el balón dependerá de diversas circunstancias [BENA13]:

- Lugar del campo donde se toma el balón
- Si hay ventaja o desventaja numérica
- Marcador del partido

7.6.2.3 Cambio de Orientación

Benarroch [BENA13] lo define como mover el balón mediante un pase largo o una secuencia, a un lugar alejado de la situación actual. Es lo que popularmente se conoce como cambio de juego, aunque es un “término que rechazan los entrenadores titulados, argumentando que el juego sigue siendo el mismo - fútbol - lo único que se cambia es la trayectoria del balón” [BENA13].

7.6.2.4 Paredes

Es una jugada ofensiva muy gustada en el fútbol (aunque en nuestro medio son difícilmente bien ejecutadas).

Benarroch [BENA13] lo define como:

> Es la acción técnico – táctica colectiva ofensiva que consiste en el pase efectuado por un jugador (iniciador de la pared) sobre un compañero (receptor de la pared) y la devolución de dicho pase mediante un solo toque, bien sobre el jugador pasador, o bien, sobre otro compañero; todo ello, con el objetivo principal de desbordar a uno o varios adversarios y, con ello, superar las líneas adversarias.
>
> Las paredes son combinaciones o jugadas de acción rápida de entrega y devolución del balón entre dos o más jugadores de un equipo, en las que el jugador que recibe el balón tiene que devolverlo mediante un solo contacto, para salvar la oposición desbordando a uno o varios adversarios. [BENA13]

Este es uno de los tipos de jugadas que en un trabajo futuro se podrían implementar con técnicas de jugadas parcialmente planificadas.

7.6.2.5 El desdoblamiento

Es un concepto técnico que requiere mucha coordinación y disciplina entre los jugadores. Benarroch [BENA13] lo define como:

> Es una acción ofensiva, con carácter defensivo, que se realiza entre dos o más jugadores del equipo que posee el balón y que consiste en intercambiar sus posiciones y, por tanto, sus misiones dentro del sistema de

juego, permitiendo cubrir los espacios libres dejados por el jugador que se ha sumado al ataque.

Cuando un jugador, quien ocupa un sitio más atrasado, deja su lugar y misión para incorporarse al ataque y rebasa a otro jugador de una línea más avanzada, el jugador rebasado debe "bajar" para ocupar la posición y las funciones del que lo superó.

Ilustración 7.15: Desdoblamientos [BENA13]

Es decir, un jugador de una posición más adelantada (en zona de acción ofensiva) ocupa la posición de su compañero (en zona de acción defensiva) en una posición más atrasada, la cual abandonó al rebasarlo e incorporarse al ataque. [BENA13]

7.6.2.6 Desmarque

Es una acción básica para el ataque. Consiste en alejarse de los jugadores contrarios para poder facilitar un pase. Además Benarroch indica que “El jugador que se desmarca atrae la atención de los adversarios, lo que facilita el trabajo de quien lleva el balón, determinando la dirección, la manera y el momento oportuno del pase”[BENA13].

8 . Resultados de Investigación

8.1 Aspectos Generales

Esta investigación se centra en el diseño del “cerebro” del Agente Inteligente de Robocopa. Se ha decidido diseñarlo de manera que los agentes cumplan con las características dadas por Julián et al. [JUBO00] de ser reactivo, pro-activo y social.

El componente pro-activo se implementará mediante las decisiones que debe tomar el jugador cuando:

1. Su equipo no está en posesión del balón, entonces deberá buscar realizar una acción que ayude a sus objetivos y a los del grupo.
2. Tiene posesión del balón y debe actuar de acuerdo con su misión en el juego, dependiendo de la posición en el equipo.

El componente reactivo se acciona principalmente cuando pierde el balón (debe reaccionar para cambiar a modo de defensa) y cuando el equipo recupera el balón (debe reaccionar para cambiar al modo de ataque).

El componente social, no se aborda en detalle en esta investigación; sin embargo, sí se harán recomendaciones al respecto para hacer posible la implementación de la lógica propuesta.

8.2 Antecedentes y Experimentos iniciales

Como se mostró en la revisión de literatura, ya se han hecho trabajos para incorporar redes neuronales en jugadores de Robocopa, generalmente para tomar una decisión específica. Una aproximación que se toma como una referencia especial para esta investigación es la de Hossein et al. [HOKA05], donde se muestra cómo implementar una red para la decisión de si realizar un remate a marco o no y de efectuarse a qué área del marco rematar.

Además el modelo utilizado por Kalyviotis y Hu [KAHU02] para decidir cuál de las jugadas predefinidas para realizar un saque de puerta da una buena idea de la utilidad de redes neuronales simples de retro-propagación para modelar los problemas de toma de decisión que enfrenta un jugador de fútbol.

Como experimento inicial se realizó una sencilla prueba de la utilización de las redes neuronales, para la toma de decisiones de un agente de fútbol de Robocopa. La intención de la red es mejorar la toma de decisiones de un jugador de campo. La decisión a tomar es una de las siguientes:

1. rematar a marco
2. avanzar más hacia el marco
3. hacer un pase a un compañero
4. despejar (patear fuerte)

Los parámetros de entrada eran:

1. Si el jugador que toma la decisión está más cerca al marco que su compañero menos marcado.
2. Si tiene menos marca que todos sus compañeros cercanos.
3. Si está relativamente cerca del marco.

El diseño de la red, mostrado en la ilustración 8.1, sigue el mismo concepto utilizado por Hossein et al. [HOKA05] donde se dan entradas binarias y la salida (para minimizar cantidad de neuronas de salida) es de dos neuronas cada una formando un dígito de un número binario.

Para probar el desempeño del jugador utilizando la red contra la misma versión del jugador sin la red, se utilizaron mensajes de depuración. Se pudo comprobar que los jugadores utilizando la red tienden a realizar más pases que con la lógica anterior, se puede ver cómo combinan las diferentes estrategias. Se efectuó un partido entre un equipo con el jugador modificado con la red contra un equipo formado por instancias del original, en la mayoría de juegos ganó el que tenía la red, por cada 10 partidos, el equipo con el nuevo jugador ganaba entre 7 y 8 partidos.

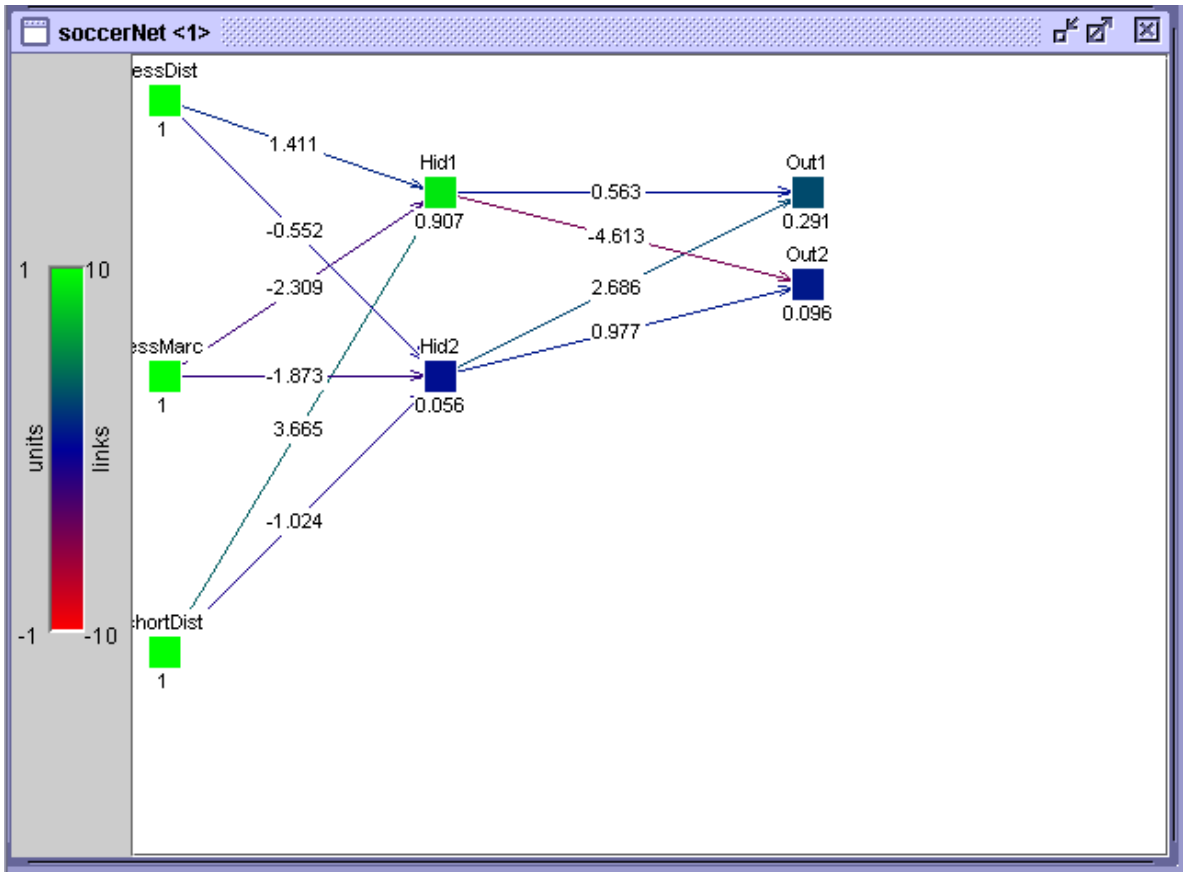


Ilustración 8.1: Red Neuronal jugador de campo

Pese a lo limitado del ejercicio anterior se puede notar que aún una red neuronal muy simple puede dar una mejoría significativa en situaciones de tomas de decisiones.

8.3 Redes Neuronales de acuerdo con el tipo de jugador

A raíz del análisis de los antecedentes y el experimento inicial, se refuerza la idea de que el desempeño de los agentes jugadores de Robocopa se puede

mejorar si una o varias redes neuronales, que se utilicen no se implementan igual para todos los jugadores, se plantea que dado que cada jugador tiene una misión distinta en el juego (de acuerdo con la posición y la guía del estratega), de manera que se pueden diseñar y entrenar redes diferentes para diferentes funciones en el juego.

Se podría pensar, por ejemplo que si se combinan las aproximaciones de Kalyviotis y Hu [KAHU02] para el saque de puerta con la de Hossein et al. [HOKA05] para los delanteros, el desempeño general del equipo debe ser mejor.

Mezclando entonces redes que se implementen en jugadas de “balón parado” junto con redes que orienten el comportamiento de los jugadores durante juego activo, se ve que un jugador de Robocopa cumple con las fases de colaboración de los agentes inteligentes descrita por Griffiths [GRIF05] a saber selección del plan, adopción de intención y acción de grupo.

De acuerdo con lo anterior se decidió hacer un segundo experimento con una red neuronal para el portero, otra para el medio y otra para el delantero.

8.3.1 Entorno del Experimento

Para probar las ideas de la red neuronal, sin tener otros factores de implementación de un jugador que pudieran distorsionar los resultados de la red, se procedió a implementar un agente jugador muy básico, el cual incluye la programación necesaria para entender los mensajes que el servidor de Robocopa envía, basándose en la información brindada por Chet et al [CHFO03], y lógica

para interpretar los objetos que ve, lo requerido para poder responder a los momentos del juego (saque de puerta, saque de banda, tiro libre, etc.) y funcionalidad para distinguir su misión (portero, defensa, medio, delantero).

Claramente este no es un agente completamente competitivo, pero sí uno que puede participar en un juego (claramente sin esperanzas de vencer, si el equipo está formado únicamente por instancias de este agente básico). Para comprobar que puede responder lo necesario, se puso a jugar contra los agentes utilizados en el experimento anterior. Los resultados, como era de esperar, fueron que se pudo terminar el encuentro, pero con marcadores adversos de 10-1, 8 – 2, etc.

Cada vez que se implementó una red para cada tipo de jugador, para evaluarla se incluyeron los reportes de bitácora (“logs”) necesarios para poder analizar el comportamiento de la red, la cual había sido previamente evaluada mediante ingreso de casos de prueba y evaluación de los resultados.

Las redes neuronales que se muestran fueron creadas usando un Applet de Distribución Libre llamado “Neural Applet” en su versión 4.3.8. Esta aplicación permite crear las neuronas con solo arrastrar y pegar y de la misma manera los enlaces. Se puede seleccionar la función de activación entre las opciones de sigmoidal, lineal, Tanh y exponencial. La aplicación permite manejar los valores de inicialización, la tasa de aprendizaje, el máximo error permitido y además luego del entrenamiento ofrece estadísticas de éxito.

8.3.2 Red Neuronal para Portero

Las decisiones que usualmente debe tomar un portero de Robocopa no son muchas, intuitivamente se puede pensar en las citadas a continuación, aunque Benarroch [BENA13] define diversas tácticas para aportar en juego ofensivo, balón parado, ir a los pies del rival, etc:

- Capturar el balón
- Avanzar
- Patear (a manera de despeje)
- Pasar el balón a un compañero
- Girar (para tener mejor visualización)
- O bien no hacer nada.

Tomando esas decisiones como base, se decidió crear una red neuronal cuyas salidas fueran las descritas anteriormente, y las entradas se determinaron que podían ser valores binarios que le dieran a la red parámetros para decidir y que estuvieran disponibles en el ambiente perceptible de la Robocopa.

Los siguientes valores fueron usados como entradas:

- Ve Marco: que indica si el portero está de frente a su portería (1) o si está de espaldas (0).
- Salido en Área: este valor indica si el portero está salido pero dentro del

área grande (1), de manera que puede capturar el balón. Si el valor está en 0 indica que no se cumple que esté salido o bien está salido, pero fuera del área.

- Fuera de Área: este es un valor que está en 1 únicamente si el portero está fuera de su área (no puede capturar el balón).
- Amigo: indica si el portero tiene a un amigo cercano al cual le puede dar el balón.
- Balón: indica si el balón es visible y está a una distancia capturable.
- Balón Atrás: le indica al portero si tiene el balón a sus espaldas..
- Contrario: le dice al portero si tiene contrarios cerca, de manera que puede haber peligro.

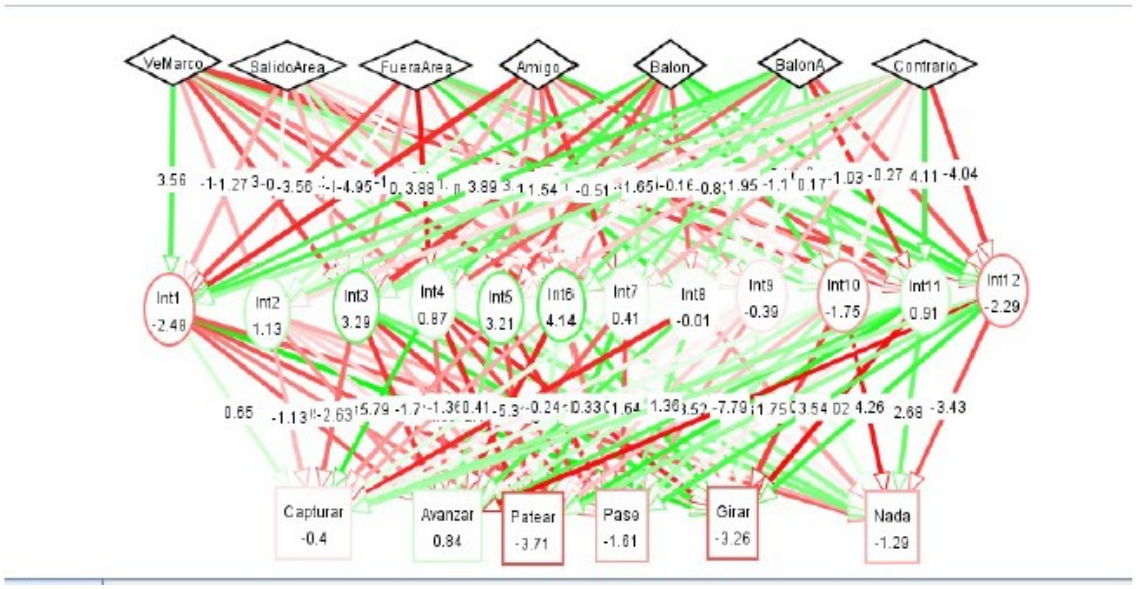


Ilustración 8.2: Red Neuronal de Portero en Neural Applet

El entrenamiento de la red se dio en 1781 ciclos. El conjunto de entrenamiento era pequeño dado que de un total de 72 casos de entrada posibles (filtrando los que no tenían sentido, como que estuviera salido en Área y Fuera de Área), se entrenó con 42. Luego se probó con 28 casos, de los cuales predijo correctamente todos. Es importante que no todos los casos con los que se probó fueron utilizados durante el entrenamiento. El conjunto de entrenamiento se puede observar en detalle en el Apéndice #1.

Para el aprendizaje se usó una tasa de aprendizaje de 0,2 y un error esperado de 0,1.

La siguiente ilustración muestra el gráfico del error total durante el entrenamiento:

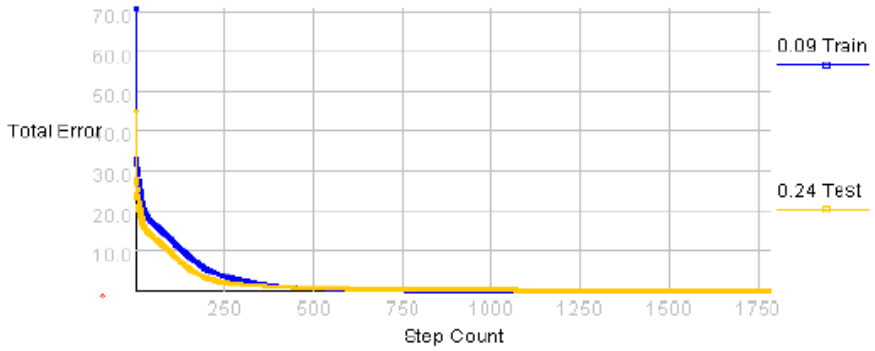


Ilustración 8.3: Error durante entrenamiento

Luego de codificar la red neuronal en el agente básico de prueba, se pudo ver tomando las decisiones correctas, sin embargo, el resultado en el juego no fue satisfactorio. Luego de las evaluaciones y revisiones de los archivos de bitácora, se pudo ver que el portero necesita mayor lógica para poder detectar las

situaciones de juego de manera más eficaz, pues usualmente cuando la red arrojaba que debía capturar el balón, al momento de ejecutar la acción ya era tarde.

8.3.3 Red Neuronal para Defensa

Buscando abordar el problema desde una perspectiva simple, se puede pensar que las decisiones que un jugador en su rol de defensa, puede tomar, una vez que se tiene el balón se pueden limitar a:

- Despeje: o bien patear hacia adelante.
- Pasar el balón
- Rematar: pues un defensa puede estar en situación de anotar gol.
- Dar un pequeño giro para mejorar ubicación y mejor visión.
- Dar un giro grande: para responder a situaciones de juego en que tenga que cambiar de dirección o bien para observar mejor el juego actual.
- Avanzar: simplemente moverse hacia la dirección actual.
- Volver: que sería buscar su zona definida por estrategia.
- Patear hacia atrás: en caso de que necesite despejar aún cuando esté viendo hacia su propio marco.
- Nada: para reflejar situación en que el juego no le exige abandonar su posición actual.

De esta manera se buscó implementar una red que tuviera dichas decisiones como salida y recibiendo como entrada:

- Marco Propio: si está viendo su propio marco.
- Ve Balón: si ve el propio balón.
- Compañero: si tiene a un compañero cerca.
- Pateable: si el balón está en una distancia donde pueda patearlo.
- Propia A: si ve su propia área
- Línea T: si ve la línea que marca el límite de la banda superior del campo de juego.
- Línea B: si ve la línea que marca el límite de la banda inferior del campo de juego.
- Centro T: si ve el centro de la línea superior del campo.
- Centro: si ve el centro del campo.
- Centro B: si ve el centro de la línea inferior del campo.
- Área contraria: si ve el área contraria.
- Marco contrario: si ve el marco contrario.

El entrenamiento de esta red, probó no ser tan simple como otras, principalmente por la cantidad de conexiones que están involucradas; cada ciclo toma mucho tiempo.

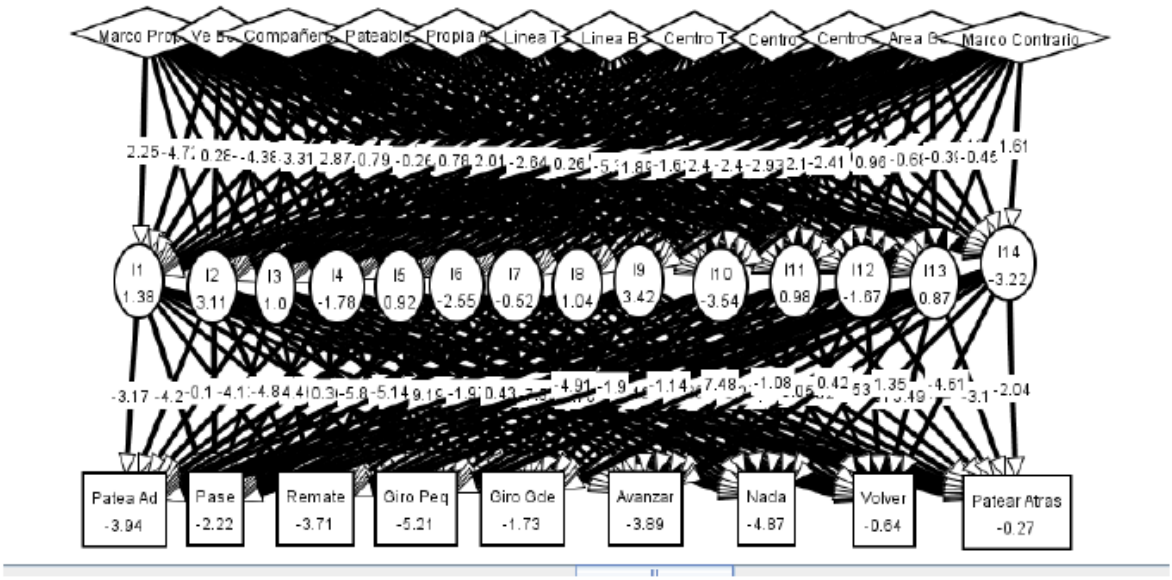


Ilustración 8.4: Primer aproximación red defensa

Los resultados del entrenamiento tampoco fueron los óptimos, no se logró convergencia y luego de más de 25,000 ciclos de entrenamiento, el error estaba estacionado en 15,0.

Se probó eliminar nodos intermedios, esto hizo el proceso más rápido hasta lograr alcanzar 19,57 de error, de ahí en adelante el progreso fue lento nuevamente. Además se intentó con tasas de aprendizaje desde 0,1 a 0,5. En la siguiente figura se muestra un gráfico de como el error deja de disminuir.

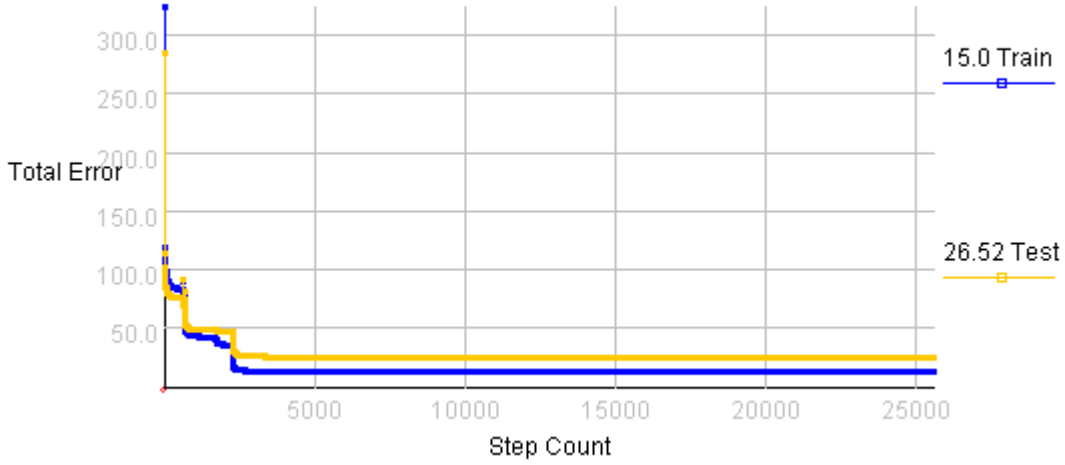


Ilustración 8.5: Error estancado en entrenamiento

Analizando los datos se encontró que de los 218 casos posibles existentes en el conjunto de datos, únicamente 3 utilizaban el giro grande, por lo cual se decidió convertir Giro Pequeño y Giro Grande en un solo nodo Giro. El resultado al entrenar fue similar. Los datos anteriores se consiguieron con 156 casos de entrenamiento; sin embargo al elevar los casos a 170 se mantenía la misma situación, deteniéndose el aprendizaje en 14,02 luego de 27033 ciclos.

Luego de analizar las posibles entradas y salidas se nota que es el mismo conjunto de entrenamiento lo que causa dificultades al entrenar la red, pues la decisión de cuándo ejecutar una acción u otra es en algunos casos no muy marcada, pudiendo hacer que no hayan suficientes elementos para clasificarlo de una manera u otra, entonces se puede causar problemas en la retro-propagación dado que al entrenar para un caso, pueda causar un efecto negativo en otros casos. Esto se nota en que el error en la herramienta se ve bajar y subir de un

ciclo a otro.

Ejemplos como el siguiente pueden provocar el problema mencionado anteriormente:

Entrada (valores en cero no mencionados)	Salida
Ve Balón=1	Patea AD
Compañero=1	
Pateable=1	
Propia A=1	
Centro=1	

Entrada (valores en cero no mencionados)	Salida
Ve Balón=1	Pase
Compañero=1	
Pateable=1	
Propia A=1	
Centro T=1	

En la comparación anterior, la única diferencia la hace si ve el centro del campo o el centro del campo pero en la línea superior (hacia una banda) y eso cambia la decisión de si hacer un pase o patear.

Decisiones tan específicas son subjetivas e incluso deberían depender de otros factores, tales como si el equipo está atacando, va arriba en el marcador, si

su compañero tiene mucha marca, etc. Al tener varios casos como el anterior, se entiende que la red no se pueda ajustar fácilmente, al menos con el algoritmo de retro-propagación con pocos datos en el conjunto de entrenamiento, el cual por las características propias del problema no puede ser más amplio.

Aquí se enfrenta un problema apuntado anteriormente por José Castro [CAST00] y José Helo [HELO96] sobre la parálisis de la red.

Más adelante se mostrará una mejor manera de modelar las decisiones para un defensa basada en la información de Benarroch [BENA13].

8.3.4 Red Neuronal para Mediocampista y delanteros

Para las posiciones de medio campista y delanteros se creó una misma red neuronal. Se hizo el experimento utilizando la misma red neuronal pero utilizando dos conjuntos de entrenamiento levemente diferentes, una resolviendo efectivamente todos los casos y otra fallando en una ocasión.

La limitante del experimento es que el conjunto de entrenamiento, en ambos casos, es muy pequeño, esto debido a que no hay muchas variantes que se apliquen dados los parámetros de entrada definidos.

El diseño de la red utilizado en ambos casos es como se muestra en la siguiente figura.

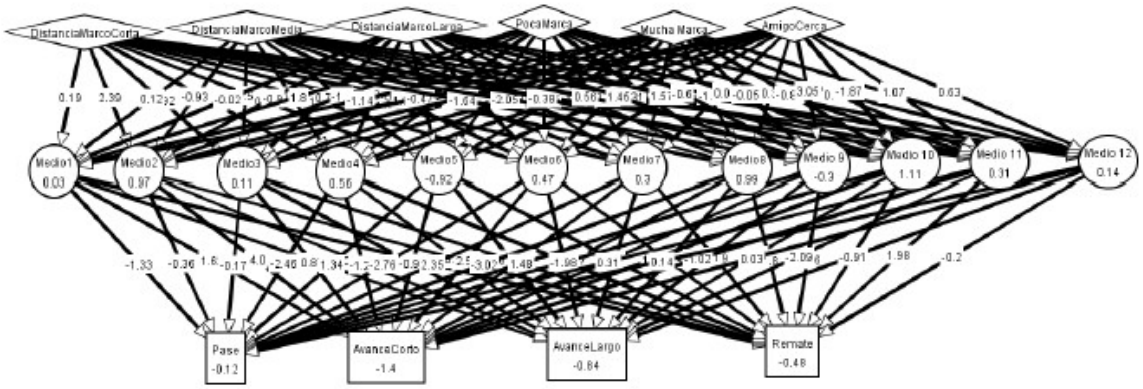


Ilustración 8.6: Red de mediocampista y delantero

El conjunto de entrenamiento en el primer caso se ve así:

Entradas:

Dist C	Dist M	Dist L	Poca M	Mucha M	Amigo Cerca
0	0	1	1	0	0
0	0	1	1	0	1
0	0	1	0	1	0
0	0	1	0	1	1
0	1	0	1	0	0
0	1	0	1	0	1
0	1	0	0	1	1
0	1	0	0	1	0
1	0	0	1	0	0
1	0	0	1	0	1
1	0	0	0	1	0
1	0	0	0	1	1

Salidas:

Pase	Avance corto	Avance Largo	Remate
0	0	1	0
0	1	0	0
0	0	1	0
1	0	0	0
0	1	0	0
0	1	0	0
1	0	0	0
0	0	1	0
0	0	0	1
0	0	0	1
0	0	0	1
1	0	0	0

Este caso predijo correctamente todos los ejemplos excepto:

0	1	0	1	0	0

el cual lo predijo como avance largo en vez de avance corto.

Dado que estos casos de ejemplo se definieron empíricamente (al igual que los del defensor) se analizaron y se encontró que se puede sufrir del mismo problema que la red del defensor, donde no hay una diferencia clara para producir un resultado u otro, haciendo que al entrenar la red, al ajustar pesos para un caso se desajusten para otro. En ésta, dado que el conjunto es más pequeño, no causa que la red no pueda converger bajo un nivel de error aceptable, además se puede identificar fácilmente la causa del problema.

En los primeros dos casos de entrenamiento se nota que uno está buscando predecir avance largo y el siguiente avance corto, pero la única diferencia entre ambos es si tiene un compañero cerca, lo cual per se no es un factor determinante para decidir si avanza poco o mucho.

Modificando únicamente el caso del avance corto o largo para entrenar para los resultados a continuación, se pudo corregir el error.

Este caso predijo bien todos los casos incluso utilizando un ejemplo menos durante el entrenamiento que en el intento anterior.

Pase	Avance corto	Avance Largo	Remate
0	1	0	0
0	1	0	0
0	0	1	0
1	0	0	0
0	1	0	0
0	1	0	0
1	0	0	0
0	0	1	0
0	0	0	1
0	0	0	1
0	0	0	1
1	0	0	0

Al implementar esta red para los medios y delanteros se nota que toman las decisiones de manera correcta, aún con las limitaciones que tiene el jugador, se ve un avance en su desempeño.

DEBUG AWT-EventQueue-0 jugador.Jugador - Jugador#10 ve balon, no pateable - correr

DEBUG AWT-EventQueue-0 jugador.Jugador - Jugador#8- realizar_accion_juego_medio_delantero

DEBUG AWT-EventQueue-0 jugador.Jugador - Jugador#8 ve balon, no pateable - girar

DEBUG AWT-EventQueue-0 jugador.Jugador - Jugador#6- realizar_accion_juego_medio_delantero

DEBUG AWT-EventQueue-0 jugador.Jugador - Jugador#6 ve balon, no pateable - correr

DEBUG AWT-EventQueue-0 jugador.Jugador - Jugador#11- realizar_accion_juego_medio_delantero

DEBUG AWT-EventQueue-0 jugador.Jugador - Jugador#11 ve balon, no pateable - correr

DEBUG AWT-EventQueue-0 jugador.Jugador - Jugador#9- realizar_accion_juego_medio_delantero

DEBUG AWT-EventQueue-0 jugador.Jugador - Jugador#9 ve balon, no pateable - girar

DEBUG AWT-EventQueue-0 jugador.Jugador - Jugador#7- realizar_accion_juego_medio_delantero

DEBUG AWT-EventQueue-0 jugador.Jugador - Jugador#7 ve balon, no pateable -

correr

DEBUG AWT-EventQueue-0 jugador.Jugador - Jugador#10-realizar_accion_juego_medio_delantero

DEBUG AWT-EventQueue-0 jugador.Jugador - Jugador#10 ve balon, no pateable - correr

DEBUG AWT-EventQueue-0 jugador.Jugador - Jugador#8-realizar_accion_juego_medio_delantero

DEBUG AWT-EventQueue-0 jugador.Jugador - Jugador#8 ve balon, no pateable - girar

DEBUG AWT-EventQueue-0 jugador.Jugador - Jugador#6-realizar_accion_juego_medio_delantero

DEBUG AWT-EventQueue-0 jugador.Jugador - Jugador#6 ve balon, no pateable - correr

DEBUG AWT-EventQueue-0 jugador.Jugador - Jugador#11-realizar_accion_juego_medio_delantero

DEBUG AWT-EventQueue-0 jugador.Jugador - Jugador#11 ve balon, no pateable - correr

DEBUG AWT-EventQueue-0 jugador.Jugador - Jugador#9-realizar_accion_juego_medio_delantero

DEBUG AWT-EventQueue-0 jugador.Jugador - Jugador#9 ve balon, no pateable - girar

DEBUG AWT-EventQueue-0 jugador.Jugador - Jugador#7-realizar_accion_juego_medio_delantero

DEBUG AWT-EventQueue-0 jugador.Jugador - Jugador#7 ve balon, no pateable - correr

DEBUG AWT-EventQueue-0 jugador.Jugador - Jugador#10-realizar_accion_juego_medio_delantero

DEBUG AWT-EventQueue-0 jugador.Jugador - Jugador #10Entradas--> 1.0 0.0 0.0 0.0 1.0 0.0

DEBUG AWT-EventQueue-0 jugador.Jugador - Jugador #10Salida --> 0.0 0.0 0.0 1.0

DEBUG AWT-EventQueue-0 jugador.Jugador - Jugador #10: remate a porteria

8.3.5 Red Neuronal para Defensa Modificada

Uno de los problemas identificados en un jugador simple de Robocopa es que tienden a correr todos detrás del balón. Las redes neuronales implementadas

anteriormente no previenen este problema de buena manera, pues los parámetros de entrada responden más a la percepción individual de los jugadores y no tomando en cuenta las circunstancias del juego y las acciones que otros compañeros puedan estar realizando.

En vista de lo mencionado anteriormente, se plantea que las decisiones básicas a tomar cuando se tiene el balón pueden ser resueltas con una red neuronal muy sencilla, similar a la presentada en Antecedentes y Experimentos iniciales o con redes similares a la utilizada para medios y delanteros con las modificaciones necesarias de acuerdo con el rol del jugador. El comportamiento propio de la misión del jugador de acuerdo con las circunstancias de juego debería implementar técnicas como las propuestas por Benarroch [BENA13], en este caso se toma preferencia por estrategias para defensa en zona.

El primer punto que debe tener claro un defensor, para poder implementar estas técnicas, es el saber la zona que le corresponde cubrir, luego es clave saber si el equipo tiene o no posesión del balón. De ahí se pueden extraer otras características o situaciones importantes para tomar su decisión. En este caso se utilizaron las siguientes:

- Modo: si está en ataque o defensa. Eso depende de si se tiene el balón.
- Zona: si está en la zona asignada o no.
- Balón: si se observa el balón.
- BalónZ: si el balón está en la zona.

- BalónMarco: si el balón está cerca del marco.
- AmigoBalón: si hay compañeros más cerca del balón.
- ContrarioBalón: si hay contendientes más cerca del balón.
- Contrario: si hay contrarios a la vista.

Basado en las entradas mostradas anteriormente, el jugador puede tomar decisiones como:

- Esperar: no hacer nada por el momento.
- Girar: para tener otra perspectiva del juego.
- IrABalón: ir hacia el balón para tenerlo en poseción..
- Cobertura: buscar hacer una cobertura a un jugador.
- Desmarcarse: alejarse de un contrario cercano..
- Apoyar: ir a auxiliar a un compañero que puede estar en apuros.
- VolverZona: retornar a la zona designada en la estrategia.

De esta manera se diseñó una red neuronal como la mostrada en la Ilustración 8.7.

Se intentó entrenar esta red con un conjunto máximo de entrenamiento de 256 casos de los cuales se descartaron 121 por corresponder a situaciones que no son factibles, tales como no ver el balón y saber que el balón está en la zona, pues son contradictorias, quedando un conjunto total de 135. De este conjunto se

tomaron primeramente 91 casos para entrenamiento, contra 60 para pruebas.

Tras más de 20,000 ciclos de entrenamiento, la red no llegó a converger, se notó una fuerte desaceleración del aprendizaje luego de los primeros 15,000 ciclos.

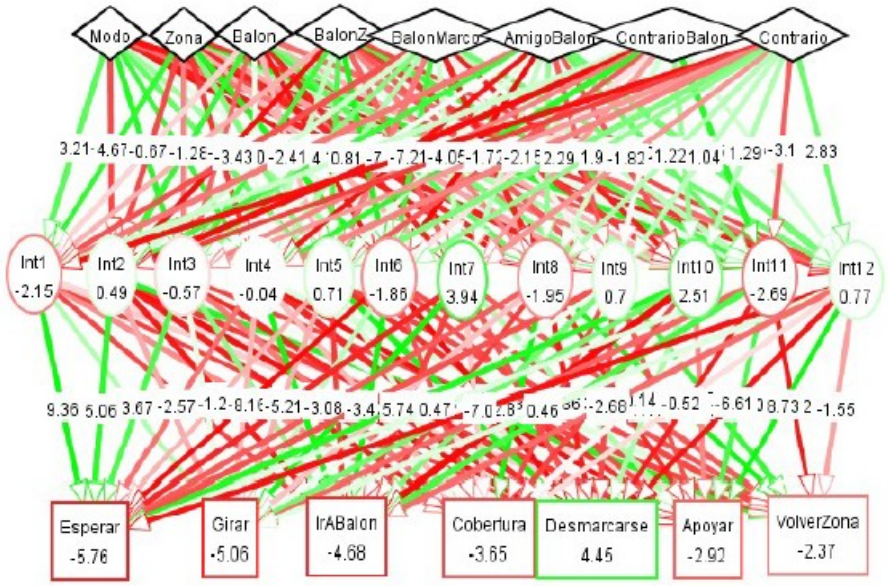


Ilustración 8.7: Red Neuronal defensa inspirado por técnicas de Benarroch [BENA13]

Siguiendo lo sugerido por Goni [GORN97] se eleva la cantidad de casos de entrenamiento al 80%, de esta manera se procede a entrenar con 97 casos. En esta ocasión la red entrenó en 3479 ciclos, dando un error de 0,1. De 60 casos con los cuales se probó, únicamente falló en 1, para un 98% de efectividad.

Además pruebas realizadas con casos no incluidos ni en el conjunto de entrenamiento ni en el conjunto de prueba, dieron buenos resultados prediciendo bien el resultado en 26 de 37.

Los detalles de los datos usados para entrenar, probar durante

entrenamiento y casos no incluidas del todo durante el entrenamiento se puede ver el Apéndice #2.

9 . Conclusiones y Recomendaciones

Luego de haber realizado los diversos ensayos de esta investigación se han formulado las conclusiones y recomendaciones que se muestran a continuación.

9.1 Conclusiones

Para ubicar al lector con más facilidad, se agrupan las conclusiones por cada uno de los objetivos específicos.

- **Objetivo 1**:

  Comprender el ambiente de Robocopa y las decisiones importantes que debe tomar un agente.

  1. Una vez examinado el ambiente de Robocopa, incluyendo los datos que el servidor provee en cada ciclo al cliente, y que el jugador únicamente puede realizar una acción por ciclo, las acciones que tome el jugador deben estar basadas en aspectos que le permitan seguir la estrategia de acuerdo con su misión (o posición de juego), tales como:

     - Cantidad de marca.
     - Cercanía o lejanía del balón.
     - Cercanía o lejanía de compañeros y contrincantes.
     - Situación de juego (ataque o defensa).

2. La decisión o acción que ejecutará el jugador debe basarse en su misión en el juego, tal como se busca en los experimentos hechos en la sección Redes Neuronales de acuerdo con el tipo de jugador.
3. La acción a tomar está delimitada por lo que el ambiente de Robocopa provee, las cuales se muestran en la Ilustración 7.3 Acciones del jugador [CHFO03].

- **Objetivo 2**:

  Definir la especialización básica necesaria para diferenciar el comportamiento de agentes de Robocopa.

  1. La especialización básica para agentes de fútbol es de acuerdo con las posiciones más tradicionales del fútbol. Portero, defensa, medio campista y delantero.
  2. Para este experimento fue suficiente con especializar el comportamiento para portero, defensa y un mismo comportamiento para medio campistas y delanteros.

- **Objetivo 3**:

  Identificar escenarios que provean ejemplos de decisiones importantes para jugadores

  1. Los escenarios que hacen que el jugador deba tomar decisiones importantes van a depender de la función del jugador. Así que si el jugador está defendiendo se puede enfrentar a una situación donde

deba hacer una acción que lo lleve a una cobertura, marcaje o permuta.

2. En la sección Redes Neuronales de acuerdo con el tipo de jugador, se muestra que las redes neuronales hechas buscan que el jugador reaccione a diversos escenarios, por ejemplo en el caso del defensa, el principal disparador para que tome una decisión es si el balón está en su propia zona o si está cerca del marco propio.

- **Objetivo 4**:

  Definir las redes neuronales necesarias para especializar el comportamiento de los jugadores.

  1. Para el alcance definido en esta investigación se generaron redes neuronales para portero, defensa y una misma para medio campista y delantero.
  2. Cuando sea necesario se puede especializar más el tipo de jugador, únicamente hace falta incluir lógica en la programación para que identifique su misión y se genera una red neuronal, que atienda las necesidades básicas del rol asignado. Así mismo se debe diseñar el conjunto de entrenamiento adecuado.

- **Objetivo 5**:

  Definir la estructura de cada red neuronal necesaria.

  1. En este experimento se utilizaron redes neuronales de tres capas,

las cuales dieron buenos resultados. En los diversos casos no se encontró mayor diferencia al agregar o eliminar nodos intermedios. La cantidad se buscó que se aproximara a la suma de nodos de entrada más los nodos de salida. En el caso de la primer red entrenada para defensa se encontró que el aprendizaje era muy lento aún con menos cantidad de nodos intermedios, en todos los otros casos la cantidad fue suficiente.

2. La cantidad de nodos intermedios no agrega mucho al hacer que una red converja o no, la calidad y cantidad del conjunto de entrenamiento es más relevante.

- **Objetivo 6**:

  Crear los conjuntos de entrenamiento que permitan un buen desempeño de cada red.

  1. Para redes pequeñas y de decisiones no muy complejas no hace falta conjuntos de entrenamiento grandes. La red utilizada para el portero tiene 72 casos posibles de entrenamiento y se entrenó con 42. La red de defensa modificada tiene un total de 135 casos posibles de los cuales se entrenó con 97, es importante que cuando se intentó entrenarla con solo 91 no llegó a converger bien. La red para medio campista y delanteros es sumamente simple pues tiene muy pocas entradas, su conjunto máximo es únicamente 12 casos posibles, de los cuales se entrenó con 9.

2. En la literatura revisada se puede encontrar que para redes más complejas, para que los algoritmos funcionen bien es necesario tener grandes conjuntos de entrenamiento; esto nos indica que para hacer redes neuronales más complejas para un jugador se deben crear conjuntos de entrenamiento mayores. En los casos utilizados no fue necesario pues los factores utilizados para tomar las decisiones no son muchos.

- **Objetivo 7**:

  Evaluar las redes creadas y entrenadas extrayendo conclusiones sobre los conjuntos de entrenamiento, diseño de la red y técnicas utilizadas.

  1. Al evaluar los resultados de esta investigación se puede concluir que el diseño de la red neuronal, de acuerdo con cada tipo de jugador es una tarea prácticamente trivial, únicamente se necesita saber cuáles son los factores de entrada, cuáles son las decisiones se quiere que tome, y la cantidad de nodos intermedios se determina mediante prueba y error.
  2. Los conjuntos de entrenamiento son el insumo más importante para poder guiar el aprendizaje. Se debe buscar que sean lo más amplios posibles, pero no se debe usar muchos casos para el entrenamiento como para que la red simplemente reproduzca y no reaccione bien ante variaciones en las entradas.

3. La herramienta utilizada para entrenar las redes, hace que sea accesible para cualquiera con conocimiento básico en redes neuronales y sin mucho conocimiento matemático poder aprovechar esta técnica; sin embargo se tiene la limitación de que no se pueden hacer modificaciones en el algoritmo de aprendizaje.

9.2 Recomendaciones

Las siguientes son recomendaciones que se proponen para la realización de trabajos futuros en el tema objeto de esta investigación:

- Ampliar gradualmente la especialización de los jugadores, esto permitirá ir avanzando en lograr un comportamiento diferente para más jugadores promoviendo mayor dinamismo en el juego.

- Expander el área de toma de decisiones para manejar jugadas planificadas y jugadas parcialmente planificadas, trabajando sobre una arquitectura de agentes híbrida.

- Analizar variantes del modelo de retropropagación u otros tipos de redes neuronales buscando evitar problemas de parálisis de la red.

- Evaluar otras herramientas para creación y entrenamiento de las redes neuronales que permitan tener mayor control sobre parámetros y algoritmos de entrenamiento.

- Enriquecer los jugadores con un componente social que permita la

comunicación entre ellos, cuidando que no haya excesivo intercambio de mensajes entre los jugadores.

- Investigar en algoritmos y heurísticas adecuadas para alimentar las entradas de la red neuronal, cuyos valores no son directamente provistos por el ambiente de robocopa.

Las recomendaciones anteriores se profundizarán a manera de propuesta en el siguiente capítulo de la presente investigación.

10 . Propuestas

En el capítulo anterior se mostraron una serie de conclusiones y recomendaciones. Con el fin de aportar ideas para futuros trabajos de investigación se desarrollarán a continuación una serie de propuestas para abordar en futuras investigaciones.

10.1 Especialización de jugadores

Se debe definir cuál es la especialización deseada para los jugadores. En esta investigación se definió:

- Portero
- Defensa
- Medio campistas y delanteros

Para futuros estudios y para lograr más dinámica se propone dar un paso más en la especialización y definir:

- Portero
- Defensa
- Medios de contención
- Medios creativos
- Delanteros

Eventualmente y de acuerdo con los intereses del investigador se pueden especializar aún más los jugadores hasta un extremo de que cada jugador tenga

un comportamiento diferente de todos los demás, esto se puede ser de manera gradual.

10.2 Escenarios para toma de decisiones

Se propone ampliar los escenarios de toma decisiones para incluir jugadas planificadas y jugadas parcialmente planificadas.

Jugadas planificadas se podrían modelar para situaciones donde el balón está detenido. Es estos casos se puede recurrir a redes neuronales similares a las propuestas por Kalyviotis y Hu [KAHU02] buscando tener arquitecturas híbridas que hagan un balance entre lo reactivo y lo deliverativo como se expone en la sección Arquitecturas Híbridas.

En un juego de fútbol también pueden presentarse situaciones que se asemejen a una situación pensada con anterioridad y para la cual se puede definir una estrategia. Para modelar casos como éste se propone explorar los Posibles Mundos expuestos por Wooldridge y Jennings [WOJE95], esto haría necesario definir un método simbólico para representar las situaciones de juego, de manera que la situación actual de juego se pueda comparar rápidamente con las previstas. Esto es un reto que por sí solo amerita ser el centro de una investigación ya que se debe abordar el tema de la eficiencia, pues un jugador tiene un corto tiempo para evaluar la situación y tomar una acción apropiada. Se puede también investigar a fondo el PRS y los TouringMachines mencionados por los mismos autores para utilizar una lógica similar en cuanto a los planes parcialmente elaborados.

10.3 Creación y entrenamiento de Redes Neuronales

Para la creación de las redes neuronales se pueden experimentar con variantes del algoritmo de retropropagación, tales como los referentes a aprendizaje profundo mencionados en la sección 7.4.1.5 , especialmente si se tienen problemas para la convergencia de la red. Se pueden también implementar redes de otros tipos tales como los mencionados en la sección 7.4.2 .

Otras herramientas para la creación y entrenamiento de la red pueden ser utilizadas, pues pese a que la empleada es conveniente al ser accesible y no requiere amplia experiencia en los procesos matemáticos de los algoritmos de entrenamiento, limita la investigación al no poder tener mayor control sobre los algoritmos de entrenamiento. Además está limitada a redes neuronales de retropropagación sin retorno, de manera que no es posible implementar redes recurrentes o parcialmente recurrentes.

10.4 Comunicación entre jugadores

En esta investigación la colaboración se da como efecto emergente de las acciones individuales de los jugadores quienes buscan un objetivo común.

En futuras investigaciones se puede enriquecer el comportamiento del jugador permitiendo el intercambio de mensajes, lo cual estará limitado por el rango de audición configurado por el servidor, como se muestra en la sección 7.3 referente a Robocopa.

Se propone hacer uso de los comandos “say” y “hear” para que los jugadores se puedan transmitir mensajes, se piensa inicialmente en avisar a jugadores cuando se pierde el balón para que todos puedan cambiar rápidamente a modo de defensa, de igual manera enviar un mensaje cuando se recupera el balón, o bien cuando un jugador se siente en clara desventaja y requiere apoyo de los compañeros.

No se recomienda permitir que los jugadores abusen de la comunicación, pues de ser así los compañeros perderán todo el tiempo de su ciclo en reaccionar a los mensajes, impidiendo tomar acciones oportunas.

10.5 Algoritmos y heurísticas para alimentar las redes neuronales

Algunas entradas necesarias para las redes neuronales no están directamente provistas por el ambiente de Robocopa o pueden ser calculadas de manera trivial.

Ciertos algoritmos pueden ser mejorados para que el jugador tenga un mayor conocimiento de su situación de juego. Un ejemplo es elaborar más lo relativo al conocimiento, que tenga un jugador sobre su posición en la cancha. En este caso se abordó por las banderas que el jugador pueda ver, se puede experimentar algoritmos que asignen coordenadas en planos X y Y dando una ubicación aproximada.

En esta investigación un aspecto clave para los defensas es si su equipo

está atacando o defendiendo, y esto en realidad se puede aplicar a todos los jugadores en los que se quiera modelar su comportamiento sin el balón. Este es un tema que se propone se investigue a fondo pues para determinar si se está en ataque o defensa entran en juego diversas variables, algunas de las cuales pueden ser:

- Ubicación del balón con respecto al jugador.
- Ubicación del balón con respecto a su propio marco.
- Cercanía de compañeros con el balón.
- Cercanía de contrincantes con el balón.

Se puede incluso hacer una evaluación de variables como las mencionadas anteriormente durante un tiempo, estableciendo una memoria de la situación de juego por un lapso determinado.

11 . Apéndice #1

El conjunto de entrenamiento utilizado para la red de portero. A continuación se muestra el conjunto completo que consta de 72 casos.

Ve Marco	Salido en Area	Fuera de Area	Amigo Cerca	Ve Balón Cerca	Balón Cerca A.	Contrario Cerca	Capturar	Avanzar	Patear	Pase	Girar	Nada
0	0	0	0	0	0	0	0	0	0	0	0	1
0	0	0	0	0	0	1	0	0	0	0	0	1
0	0	0	0	0	1	0	0	0	0	0	1	0
0	0	0	0	0	1	1	0	0	0	0	1	0
0	0	0	0	1	0	0	1	0	0	0	0	0
0	0	0	0	1	0	1	1	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	0	1
0	0	0	1	0	0	1	0	0	0	0	0	1
0	0	0	1	0	1	0	0	0	0	0	1	0
0	0	0	1	0	1	1	0	0	0	0	1	0
0	0	0	1	1	0	0	1	0	0	0	0	0
0	0	0	1	1	0	1	1	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0	0	1	0
0	0	1	0	0	0	1	0	0	0	0	1	0
0	0	1	0	0	1	0	0	0	0	0	1	0
0	0	1	0	0	1	1	0	0	0	0	1	0
0	0	1	0	1	0	0	0	0	1	0	0	0
0	0	1	0	1	0	1	0	0	1	0	0	0
0	0	1	1	0	0	0	0	0	0	0	1	0
0	0	1	1	0	0	1	0	0	0	0	1	0
0	0	1	1	0	1	0	0	0	0	0	1	0
0	0	1	1	0	1	1	0	0	0	0	1	0
0	0	1	1	1	0	0	0	0	0	1	0	0
0	0	1	1	1	0	1	0	0	0	1	0	0
0	1	0	0	0	0	0	0	0	0	0	1	0
0	1	0	0	0	0	1	0	0	0	0	0	1
0	1	0	0	0	1	0	0	0	0	0	1	0
0	1	0	0	0	1	1	0	0	0	0	1	0
0	1	0	0	1	0	0	1	0	0	0	0	0
0	1	0	0	1	0	1	1	0	0	0	0	0
0	1	0	1	0	0	0	0	0	0	0	1	0
0	1	0	1	0	0	1	0	0	0	0	0	1
0	1	0	1	0	1	0	0	0	0	0	1	0
0	1	0	1	0	1	1	0	0	0	0	1	0
0	1	0	1	1	0	0	1	0	0	0	0	0
0	1	0	1	1	0	1	1	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	1	0	0	0	0	1	0
1	0	0	0	0	1	0	0	0	0	0	1	0
1	0	0	0	0	1	1	0	0	0	0	1	0
1	0	0	0	1	0	0	1	0	0	0	0	0
1	0	0	0	1	0	1	1	0	0	0	0	0
1	0	0	1	0	0	0	0	0	0	0	1	0
1	0	0	1	0	0	1	0	0	0	0	1	0
1	0	0	1	0	1	0	0	0	0	0	1	0
1	0	0	1	0	1	1	0	0	0	0	1	0
1	0	0	1	1	0	0	1	0	0	0	0	0
1	0	0	1	1	0	1	1	0	0	0	0	0
1	0	1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	0	1	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0	1	0
1	0	1	0	0	1	1	0	0	0	0	1	0
1	0	1	0	1	0	0	0	0	0	0	1	0
1	0	1	0	1	0	1	0	1	0	0	0	0
1	0	1	1	0	0	0	0	1	0	0	0	0
1	0	1	1	0	0	1	0	1	0	0	0	0
1	0	1	1	0	1	0	0	1	0	0	0	0
1	0	1	1	0	1	1	0	1	0	0	0	0
1	0	1	1	1	0	0	0	0	0	1	0	0
1	0	1	1	1	0	1	0	0	0	1	0	0
1	1	0	0	0	0	0	0	1	0	0	0	0
1	1	0	0	0	0	1	0	1	0	0	0	0
1	1	0	0	0	1	0	0	0	0	0	1	0
1	1	0	0	0	1	1	0	0	0	0	1	0
1	1	0	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	1	1	0	0	0	0	0
1	1	0	1	0	0	0	0	1	0	0	0	0
1	1	0	1	0	0	1	0	1	0	0	0	0
1	1	0	1	0	1	0	0	1	0	0	0	0
1	1	0	1	0	1	1	0	1	0	0	0	0
1	1	0	1	1	0	0	1	0	0	0	0	0
1	1	0	1	1	0	1	1	0	0	0	0	0

Para la definición del conjunto de entrenamiento se filtraron casos que claramente no son posibles, como tener Salido en Área y Fuera de Área, lo cual no tiene sentido.

Los datos de la entrada están basados en lo que un agente de Robocopa puede percibir y que es relevante para el portero.

Los primeros tres parámetros están basados en las banderas que puede percibir. Ve Marco se define si observa alguna de las banderas (f g x t), (g x), (f g x b), siendo x el lado actual del jugador (l o r). Salido en Área se define basado primeramente en una heurística a implementar por el agente para mantener su estado de si está en una posición que se considere salido o no, pero además considera si ve la línea del área grande, definida por (f p x t), (f p x c), (f p x b). Fuera de Área se define de manera similar a la anterior y son mutuamente excluyentes, no se puede estar dentro y fuera del área, tal como se explicó anteriormente.

Los siguientes parámetros de entrada se definen basados en los objetos que el agente puede percibir y herísticas adecuadas para determinar valores relativos de cercanía o lejanía.

Las acciones están definidas por lo que un portero puede hacer en Robocopa, siendo Capturar el balón (catch), avanzar (dash), patear (kick), pase (kick – hacia un compañero), girar (turn) o bien no realizar ninguna acción.

Para realizar el entrenamiento se tomaron 42 casos de los posibles 72,

buscando que hubiera representación de todas las decisiones.

A continuación se muestran los datos utilizados para el entrenamiento:

Ve Marco	Salido en Area	Fuera de Area	Amigo Cerca	Ve Balón Cerca	Balón Cerca A.	Contrario Cerca	Capturar	Avanzar	Patear	Pase	Girar	Nada
0	0	0	0	0	0	0	0	0	0	0	0	1
0	0	0	0	0	0	1	0	0	0	0	0	1
0	0	0	0	0	1	0	0	0	0	0	1	0
0	0	0	0	0	1	1	0	0	0	0	1	0
0	0	0	0	1	0	0	1	0	0	0	0	0
0	0	1	0	0	1	0	0	0	0	0	1	0
0	0	1	0	0	1	1	0	0	0	0	1	0
0	0	1	0	1	0	0	0	0	1	0	0	0
0	0	1	0	1	0	1	0	0	1	0	0	0
0	0	1	1	0	0	0	0	0	0	0	1	0
0	1	0	0	1	0	0	1	0	0	0	0	0
0	1	0	0	1	0	1	1	0	0	0	0	0
0	1	0	1	0	0	0	0	0	0	0	1	0
0	1	0	1	0	0	1	0	0	0	0	0	1
0	1	0	1	0	1	0	0	0	0	0	1	0
0	1	0	1	0	1	1	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	1	0	0	0	0	1	0
1	0	0	0	0	1	0	0	0	0	0	1	0
1	0	0	0	0	1	1	0	0	0	0	1	0
1	0	0	0	1	0	0	1	0	0	0	0	0
1	0	1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	0	1	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0	1	0
1	0	1	0	0	1	1	0	0	0	0	1	0
1	0	1	0	1	0	0	0	0	0	0	1	0
1	0	1	0	1	0	1	0	1	0	0	0	0
1	1	0	1	0	0	1	0	1	0	0	0	0
1	1	0	1	0	1	0	0	1	0	0	0	0
1	1	0	1	0	1	1	0	0	0	0	1	0
1	1	0	1	1	0	0	1	0	0	0	0	0
1	1	0	1	1	0	1	1	0	0	0	0	0
0	0	1	1	1	0	0	0	0	0	1	0	0
1	0	1	1	1	0	0	0	0	0	1	0	0
1	0	1	1	1	0	1	0	0	0	1	0	0
0	0	0	0	1	0	1	1	0	0	0	0	0
1	0	0	1	1	0	0	1	0	0	0	0	0
1	0	1	1	0	0	1	0	1	0	0	0	0
1	0	1	1	0	1	0	0	1	0	0	0	0
0	0	0	1	0	1	0	0	0	0	0	1	0
0	0	0	1	0	1	1	0	0	0	0	1	0
0	0	1	0	0	0	0	0	0	0	0	1	0

Los siguiente casos se utilizaron para probar la red.

Ve Marco	Salido en Area	Fuera de Area	Amigo Cerca	Ve Balón Cerca	Balón Cerca A.	Contrario Cerca	Capturar	Avanzar	Patear	Pase	Girar	Nada
0	0	0	0	0	0	0	0	0	0	0	0	1
0	0	0	0	0	0	1	0	0	0	0	0	1
0	0	0	0	1	0	1	1	0	0	0	0	0
0	0	0	1	1	0	1	1	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0	0	1	0
0	0	1	0	1	0	0	0	0	1	0	0	0
0	0	1	0	1	0	1	0	0	1	0	0	0
0	0	1	1	1	0	0	0	0	0	1	0	0
0	0	1	1	1	0	1	0	0	0	1	0	0
0	1	0	0	0	0	1	0	0	0	0	0	1
0	1	0	0	0	1	1	0	0	0	0	1	0
0	1	0	1	0	0	1	0	0	0	0	0	1
0	1	0	1	1	0	0	1	0	0	0	0	0
0	1	0	1	1	0	1	1	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	1	0	1	0	0	0	0	0	1	0
1	0	0	1	0	1	1	0	0	0	0	1	0
1	0	0	1	1	0	0	1	0	0	0	0	0
1	0	1	0	0	0	1	0	1	0	0	0	0
1	0	1	0	0	1	1	0	0	0	0	1	0
1	0	1	1	0	1	0	0	1	0	0	0	0
1	0	1	1	0	1	1	0	1	0	0	0	0
1	0	1	1	1	0	0	0	0	0	1	0	0
1	1	0	0	0	1	1	0	0	0	0	1	0
1	1	0	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	1	1	0	0	0	0	0
1	1	0	1	1	0	0	1	0	0	0	0	0
1	1	0	1	1	0	1	1	0	0	0	0	0

12 . Apéndice #2

El conjunto utilizado para entrenar la red de defensa con características para buscar implementar técnicas expuestas por Benarroch [BENA13] se muestra a continuación.

El conjunto con los 135 casos totales está dado por:

Modo	Zona	Balon	BalonZ	BalonMarco	AmigoBalon	ContrarioBalon	Contrario	Esperar	Girar	IrABalon	Cobertura	Desmarcarse	Apoyar	VolverZona
0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	1	0	1	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	1	0	0	0	0	0	0	1
0	0	1	0	0	0	1	0	0	0	0	0	0	0	1
0	0	1	0	0	0	1	1	0	0	0	0	0	0	1
0	0	1	0	0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	1	0	1	0	0	0	0	0	0	1
0	0	1	0	0	1	1	0	0	0	0	0	0	0	1
0	0	1	0	0	1	1	1	0	0	0	0	0	0	1
0	0	1	0	1	0	0	0	0	0	0	0	0	0	1
0	0	1	0	1	0	0	1	0	0	0	0	0	0	1
0	0	1	0	1	0	1	0	0	0	1	0	0	0	0
0	0	1	0	1	0	1	1	0	0	1	0	0	0	0
0	0	1	0	1	1	0	0	0	0	0	0	0	1	0
0	0	1	0	1	1	0	1	0	0	0	0	0	0	1
0	0	1	0	1	1	1	0	0	0	0	0	0	1	0
0	0	1	0	1	1	1	1	0	0	0	0	0	1	0
0	0	1	1	0	0	0	0	0	0	1	0	0	0	0
0	0	1	1	0	0	0	1	0	0	1	0	0	0	0
0	0	1	1	0	0	1	0	0	0	1	0	0	0	0
0	0	1	1	0	0	1	1	0	0	1	0	0	0	0
0	0	1	1	0	1	0	0	1	0	0	0	0	0	0
0	0	1	1	0	1	0	1	0	0	0	1	0	0	0
0	0	1	1	0	1	1	0	0	0	0	1	0	0	0
0	0	1	1	0	1	1	1	0	0	0	1	0	0	0
0	0	1	1	1	0	0	0	0	0	1	0	0	0	0
0	0	1	1	1	0	0	1	0	0	1	0	0	0	0
0	0	1	1	1	0	1	0	0	0	1	0	0	0	0
0	0	1	1	1	0	1	1	0	0	1	0	0	0	0
0	0	1	1	1	1	0	0	1	0	0	0	0	0	0
0	0	1	1	1	1	0	1	0	0	0	0	0	1	0
0	0	1	1	1	1	1	0	0	0	0	0	0	1	0
0	0	1	1	1	1	1	1	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	0	1	0	0	0	0	0
0	1	0	0	0	0	0	1	0	1	0	0	0	0	0
0	1	1	0	0	0	0	1	1	0	0	0	0	0	0
0	1	1	0	0	0	1	0	1	0	0	0	0	0	0
0	1	1	0	0	0	1	1	1	0	0	0	0	0	0
0	1	1	0	0	1	0	0	1	0	0	0	0	0	0
0	1	1	0	0	1	0	1	1	0	0	0	0	0	0
0	1	1	0	0	1	1	0	1	0	0	0	0	0	0
0	1	1	0	0	1	1	1	1	0	0	0	0	0	0
0	1	1	0	1	0	0	0	0	0	1	0	0	0	0
0	1	1	0	1	0	0	1	0	0	1	0	0	0	0
0	1	1	0	1	0	1	0	0	0	1	0	0	0	0
0	1	1	0	1	0	1	1	0	0	1	0	0	0	0
0	1	1	0	1	1	0	0	1	0	0	0	0	0	0
0	1	1	0	1	1	0	1	1	0	0	0	0	0	0

Modo 0	Zona 1	Balon 1	BalonZ 0	BalonMarco 1	AmigoBalon 1	ContrarioBalon 1	Contrario 0	Esperar 0	Girar 0	IrABalon 0	Cobertura 0	Desmarcarse 0	Apoyar 1	VolverZona 0
0	1	1	0	1	1	1	1	0	0	0	0	0	1	0
0	1	1	1	0	0	0	0	0	0	1	0	0	0	0
0	1	1	1	0	0	0	1	0	0	1	0	0	0	0
0	1	1	1	0	0	1	0	0	0	1	0	0	0	0
0	1	1	1	0	0	1	1	0	0	1	0	0	0	0
0	1	1	1	0	1	0	0	0	0	0	0	1	0	0
0	1	1	1	0	1	0	1	0	0	0	0	1	0	0
0	1	1	1	0	1	1	0	0	0	0	0	1	0	0
0	1	1	1	0	1	1	1	0	0	0	0	1	0	0
0	1	1	1	1	0	0	0	0	0	1	0	0	0	0
0	1	1	1	1	0	0	1	0	0	1	0	0	0	0
0	1	1	1	1	0	1	0	0	0	1	0	0	0	0
0	1	1	1	1	0	1	1	0	0	1	0	0	0	0
0	1	1	1	1	1	0	0	0	0	1	0	0	0	0
0	1	1	1	1	1	0	1	0	0	1	0	0	0	0
0	1	1	1	1	1	1	0	0	0	1	0	0	0	0
0	1	1	1	1	1	1	1	0	0	1	0	0	0	0
1	0	0	0	0	0	0	0	0	1	0	0	0	0	0
1	0	0	0	0	0	0	1	0	1	0	0	0	0	0
1	0	1	0	0	0	0	0	1	0	0	0	0	0	0
1	0	1	0	0	0	0	1	0	0	0	0	1	0	0
1	0	1	0	0	0	1	0	1	0	0	0	0	0	0
1	0	1	0	0	0	1	1	0	0	0	0	1	0	0
1	0	1	0	0	1	0	0	1	0	0	0	0	0	0
1	0	1	0	0	1	0	1	0	0	0	0	1	0	0
1	0	1	0	0	1	1	0	0	0	1	0	0	0	0
1	0	1	0	0	1	1	1	0	0	0	0	1	0	0
1	0	1	0	1	0	0	0	1	0	0	0	0	0	0
1	0	1	0	1	0	0	1	0	0	1	0	0	0	0
1	0	1	0	1	0	1	0	0	0	1	0	0	0	0
1	0	1	0	1	0	1	1	0	0	1	0	0	0	0
1	0	1	0	1	1	0	0	1	0	0	0	0	0	0
1	0	1	0	1	1	0	1	1	0	0	0	0	0	0
1	0	1	0	1	1	1	0	0	0	1	0	0	0	0
1	0	1	0	1	1	1	1	0	0	1	0	0	0	0
1	0	1	1	0	0	0	0	0	0	1	0	0	0	0
1	0	1	1	0	0	0	1	0	0	1	0	0	0	0
1	0	1	1	0	0	1	0	0	0	1	0	0	0	0
1	0	1	1	0	0	1	1	0	0	1	0	0	0	0
1	0	1	1	0	1	0	0	1	0	0	0	0	0	0
1	0	1	1	0	1	0	1	0	0	0	0	1	0	0
1	0	1	1	0	1	1	0	0	0	1	0	0	0	0
1	0	1	1	0	1	1	1	0	0	1	0	0	0	0
1	0	1	1	1	0	0	0	0	0	1	0	0	0	0
1	0	1	1	1	0	0	1	0	0	1	0	0	0	0
1	0	1	1	1	0	1	0	0	0	1	0	0	0	0
1	0	1	1	1	0	1	1	0	0	1	0	0	0	0
1	0	1	1	1	1	0	0	1	0	0	0	0	0	0
1	0	1	1	1	1	0	1	0	0	0	0	1	0	0
1	0	1	1	1	1	1	0	0	0	1	0	0	0	0
1	0	1	1	1	1	1	1	0	0	1	0	0	0	0
1	1	0	0	0	0	0	0	0	1	0	0	0	0	0
1	1	0	0	0	0	0	1	0	1	0	0	0	0	0
1	1	1	0	0	0	0	0	1	0	0	0	0	0	0
1	1	1	0	0	0	0	1	0	0	0	0	1	0	0
1	1	1	0	0	0	1	0	1	0	0	0	0	0	0
1	1	1	0	0	0	1	1	1	0	0	0	0	0	0
1	1	1	0	0	1	0	0	1	0	0	0	0	0	0
1	1	1	0	0	1	0	1	0	0	0	0	1	0	0
1	1	1	0	0	1	1	0	1	0	0	0	0	0	0
1	1	1	0	0	1	1	1	0	0	0	0	1	0	0
1	1	1	0	1	0	0	0	0	0	1	0	0	0	0
1	1	1	0	1	0	0	1	0	0	1	0	0	0	0
1	1	1	0	1	0	1	0	0	0	1	0	0	0	0
1	1	1	0	1	0	1	1	0	0	1	0	0	0	0
1	1	1	0	1	1	0	0	1	0	0	0	0	0	0
1	1	1	0	1	1	0	1	0	0	0	0	1	0	0
1	1	1	0	1	1	1	0	1	0	0	0	0	0	0
1	1	1	0	1	1	1	1	0	0	0	0	1	0	0
1	1	1	1	0	0	0	0	0	0	1	0	0	0	0
1	1	1	1	0	0	0	1	0	0	1	0	0	0	0
1	1	1	1	0	0	1	0	0	0	1	0	0	0	0
1	1	1	1	0	0	1	1	0	0	1	0	0	0	0
1	1	1	1	0	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	1	0	1	0	0	1	0	0	0	0
1	1	1	1	0	1	1	0	0	0	1	0	0	0	0
1	1	1	1	0	1	1	1	0	0	1	0	0	0	0
1	1	1	1	1	0	0	0	0	0	1	0	0	0	0
1	1	1	1	1	0	0	1	0	0	1	0	0	0	0
1	1	1	1	1	0	1	0	0	0	1	0	0	0	0
1	1	1	1	1	0	1	1	0	0	1	0	0	0	0
1	1	1	1	1	1	0	0	0	0	1	0	0	0	0
1	1	1	1	1	1	0	1	0	0	1	0	0	0	0
1	1	1	1	1	1	1	0	0	0	1	0	0	0	0
1	1	1	1	1	1	1	1	0	0	1	0	0	0	0

El conjunto utilizado para entrenar fue:

Modo	Zona	Balon	BalonZ	BalonMarco	AmigoBalon	ContrarioBalon	Contrario	Esperar	Girar	IrABalon	Cobertura	Desmarcarse	Apoyar	VolverZona
0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	1	0	1	0	0	0	0	0
0	0	1	0	0	0	0	1	0	0	0	0	0	0	1
0	0	1	0	0	0	1	0	0	0	0	0	0	0	1
0	0	1	0	0	0	1	1	0	0	0	0	0	0	1
0	0	1	0	0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	1	0	1	0	0	0	0	0	0	1
0	0	1	0	0	1	1	0	0	0	0	0	0	0	1
0	0	1	0	1	0	0	0	0	0	0	0	0	0	1
0	0	1	0	1	0	0	1	0	0	0	0	0	0	1
0	0	1	0	1	0	1	0	0	0	1	0	0	0	0
0	0	1	0	1	0	1	1	0	0	1	0	0	0	0
0	0	1	0	1	1	0	0	0	0	0	0	0	1	0
0	0	1	0	1	1	0	1	0	0	0	0	0	0	1
0	0	1	0	1	1	1	1	0	0	0	0	0	1	0
0	0	1	1	0	0	0	0	0	0	1	0	0	0	0
0	0	1	1	0	0	1	0	0	0	1	0	0	0	0
0	0	1	1	0	0	1	1	0	0	1	0	0	0	0
0	0	1	1	0	1	0	0	1	0	0	0	0	0	0
0	0	1	1	0	1	0	1	0	0	0	1	0	0	0
0	0	1	1	0	1	1	0	0	0	0	1	0	0	0
0	0	1	1	0	1	1	1	0	0	0	1	0	0	0
0	0	1	1	1	0	1	0	0	0	1	0	0	0	0
0	0	1	1	1	1	0	0	1	0	0	0	0	0	0
0	0	1	1	1	1	0	1	0	0	0	0	0	1	0
0	0	1	1	1	1	1	1	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	1	0	0	0	0	0	0
0	1	1	0	0	1	0	0	1	0	0	0	0	0	0
0	1	1	0	0	1	0	1	1	0	0	0	0	0	0
0	1	1	0	0	1	1	0	1	0	0	0	0	0	0
0	1	1	0	1	0	0	0	0	0	1	0	0	0	0
0	1	1	0	1	0	1	0	0	0	1	0	0	0	0
0	1	1	0	1	0	1	1	0	0	1	0	0	0	0
0	1	1	0	1	1	0	0	1	0	0	0	0	0	0
0	1	1	0	1	1	1	0	0	0	0	0	0	1	0
0	1	1	1	0	0	0	1	0	0	1	0	0	0	0
0	1	1	1	0	0	1	0	0	0	1	0	0	0	0
0	1	1	1	0	0	1	1	0	0	1	0	0	0	0
0	1	1	1	0	1	0	0	0	0	0	0	1	0	0
0	1	1	1	0	1	0	1	0	0	0	0	1	0	0
0	1	1	1	0	1	1	0	0	0	0	0	1	0	0
0	1	1	1	0	1	1	1	0	0	0	0	1	0	0
0	1	1	1	1	0	1	0	0	0	1	0	0	0	0
0	1	1	1	1	1	0	0	0	0	1	0	0	0	0
0	1	1	1	1	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	0	0	1	0	0	0	0	0
1	0	1	0	0	0	0	0	1	0	0	0	0	0	0
1	0	1	0	0	0	0	1	0	0	0	0	1	0	0
1	0	1	0	0	0	1	0	1	0	0	0	0	0	0
1	0	1	0	0	0	1	1	0	0	0	0	1	0	0
1	0	1	0	0	1	1	0	0	0	1	0	0	0	0
1	0	1	0	0	1	1	1	0	0	0	0	1	0	0
1	0	1	0	1	0	0	0	1	0	0	0	0	0	0
1	0	1	0	1	0	0	1	0	0	1	0	0	0	0
1	0	1	0	1	0	1	0	0	0	1	0	0	0	0
1	0	1	0	1	0	1	1	0	0	1	0	0	0	0
1	0	1	0	1	1	0	1	1	0	0	0	0	0	0
1	0	1	0	1	1	1	0	0	0	1	0	0	0	0
1	0	1	0	1	1	1	1	0	0	1	0	0	0	0
1	0	1	1	0	0	0	0	0	0	1	0	0	0	0
1	0	1	1	0	0	1	0	0	0	1	0	0	0	0
1	0	1	1	0	0	1	1	0	0	1	0	0	0	0
1	0	1	1	0	1	0	1	0	0	0	0	1	0	0
1	0	1	1	0	1	1	1	0	0	1	0	0	0	0
1	0	1	1	1	0	0	0	0	0	1	0	0	0	0
1	0	1	1	1	0	0	1	0	0	1	0	0	0	0
1	0	1	1	1	0	1	1	0	0	1	0	0	0	0

Modo	Zona	Balon	BalonZ	BalonMarco	AmigoBalon	ContrarioBalon	Contrario	Esperar	Girar	IrABalon	Cobertura	Desmarcarse	Apoyar	VolverZona
1	0	1	1	1	1	0	1	0	0	0	0	1	0	0
1	0	1	1	1	1	1	0	0	0	1	0	0	0	0
1	0	1	1	1	1	1	1	0	0	1	0	0	0	0
1	1	0	0	0	0	0	0	0	1	0	0	0	0	0
1	1	0	0	0	0	0	1	0	1	0	0	0	0	0
1	1	1	0	0	0	0	0	1	0	0	0	0	0	0
1	1	1	0	0	0	0	1	0	0	0	0	1	0	0
1	1	1	0	0	0	1	0	1	0	0	0	0	0	0
1	1	1	0	0	1	0	0	1	0	0	0	0	0	0
1	1	1	0	0	1	0	1	0	0	0	0	1	0	0
1	1	1	0	0	1	1	1	0	0	0	0	1	0	0
1	1	1	0	1	0	0	0	0	0	1	0	0	0	0
1	1	1	0	1	0	0	1	0	0	1	0	0	0	0
1	1	1	0	1	0	1	0	0	0	1	0	0	0	0
1	1	1	0	1	0	1	1	0	0	1	0	0	0	0
1	1	1	0	1	1	0	0	1	0	0	0	0	0	0
1	1	1	0	1	1	1	1	0	0	0	0	1	0	0
1	1	1	1	0	0	0	0	0	0	1	0	0	0	0
1	1	1	1	0	0	1	0	0	0	1	0	0	0	0
1	1	1	1	0	0	1	1	0	0	1	0	0	0	0
1	1	1	1	0	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	1	0	1	0	0	1	0	0	0	0
1	1	1	1	1	0	0	0	0	0	1	0	0	0	0
1	1	1	1	1	0	1	1	0	0	1	0	0	0	0
1	1	1	1	1	1	0	0	0	0	1	0	0	0	0
1	1	1	1	1	1	0	1	0	0	1	0	0	0	0
1	1	1	1	1	1	1	0	0	0	1	0	0	0	0
1	1	1	1	1	1	1	1	0	0	1	0	0	0	0

Para las pruebas, durante el entrenamiento se utilizó:

Modo	Zona	Balon	BalonZ	BalonMarco	AmigoBalon	ContrarioBalon	Contrario	Esperar	Girar	IrABalon	Cobertura	Desmarcarse	Apoyar	VolverZona
0	0	0	0	0	0	0	1	0	1	0	0	0	0	0
0	0	1	0	0	0	0	1	0	0	0	0	0	0	1
0	0	1	0	0	0	1	0	0	0	0	0	0	0	1
0	0	1	0	0	1	0	1	0	0	0	0	0	0	1
0	0	1	0	0	1	1	0	0	0	0	0	0	0	1
0	0	1	0	1	0	0	0	0	0	0	0	0	0	1
0	0	1	0	1	0	1	0	0	0	1	0	0	0	0
0	0	1	0	1	0	1	1	0	0	1	0	0	0	0
0	0	1	0	1	1	0	1	0	0	0	0	0	0	1
0	0	1	1	0	0	0	0	0	0	1	0	0	0	0
0	0	1	1	0	1	0	0	1	0	0	0	0	0	0
0	0	1	1	0	1	0	1	0	0	0	1	0	0	0
0	0	1	1	1	0	1	0	0	0	1	0	0	0	0
0	0	1	1	1	0	1	1	0	0	1	0	0	0	0
0	0	1	1	1	1	1	0	0	0	0	0	0	1	0
0	1	1	0	0	1	0	1	1	0	0	0	0	0	0
0	1	1	0	0	1	1	0	1	0	0	0	0	0	0
0	1	1	0	1	0	0	0	0	0	1	0	0	0	0
0	1	1	0	1	0	1	0	0	0	1	0	0	0	0
0	1	1	0	1	1	0	0	1	0	0	0	0	0	0
0	1	1	0	1	1	0	1	1	0	0	0	0	0	0
0	1	1	1	0	0	1	0	0	0	1	0	0	0	0
0	1	1	1	0	1	0	0	0	0	0	0	1	0	0
0	1	1	1	0	1	0	1	0	0	0	0	1	0	0
0	1	1	1	0	1	1	0	0	0	0	0	1	0	0
0	1	1	1	0	1	1	1	0	0	0	0	1	0	0
0	1	1	1	1	1	1	0	0	0	1	0	0	0	0
0	1	1	1	1	1	1	1	0	0	1	0	0	0	0
1	0	1	0	0	0	0	1	0	0	0	0	1	0	0

Modo	Zona	Balon	BalonZ	BalonMarco	AmigoBalon	ContrarioBalon	Contrario	Esperar	Girar	IrABalon	Cobertura	Desmarcarse	Apoyar	VolverZona
1	0	1	0	0	1	0	0	1	0	0	0	0	0	0
1	0	1	0	0	1	1	1	0	0	0	0	1	0	0
1	0	1	0	1	0	1	1	0	0	1	0	0	0	0
1	0	1	0	1	1	0	0	1	0	0	0	0	0	0
1	0	1	1	0	0	0	0	0	0	1	0	0	0	0
1	0	1	1	0	0	1	1	0	0	1	0	0	0	0
1	0	1	1	1	0	0	0	0	0	1	0	0	0	0
1	0	1	1	1	0	0	1	0	0	1	0	0	0	0
1	0	1	1	1	0	1	1	0	0	1	0	0	0	0
1	0	1	1	1	1	0	1	0	0	0	0	1	0	0
1	0	1	1	1	1	1	0	0	0	1	0	0	0	0
1	1	0	0	0	0	0	0	0	1	0	0	0	0	0
1	1	1	0	0	0	0	0	1	0	0	0	0	0	0
1	1	1	0	0	0	0	1	0	0	0	0	1	0	0
1	1	1	0	0	1	0	0	1	0	0	0	0	0	0
1	1	1	0	0	1	0	1	0	0	0	0	1	0	0
1	1	1	0	0	1	1	0	1	0	0	0	0	0	0
1	1	1	0	1	0	0	0	0	0	1	0	0	0	0
1	1	1	0	1	0	0	1	0	0	1	0	0	0	0
1	1	1	0	1	0	1	1	0	0	1	0	0	0	0
1	1	1	0	1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	1	0	0	0	0
1	1	1	1	0	0	1	0	0	0	1	0	0	0	0
1	1	1	1	0	0	1	1	0	0	1	0	0	0	0
1	1	1	1	0	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	1	0	1	0	0	1	0	0	0	0
1	1	1	1	1	1	0	0	0	0	1	0	0	0	0
1	1	1	1	1	1	0	1	0	0	1	0	0	0	0
1	1	1	1	1	1	1	0	0	0	1	0	0	0	0
1	1	1	1	1	1	1	1	0	0	1	0	0	0	0

Los siguientes casos no fueron incluídos de ninguna manera, ni como entrenamiento ni como prueba durante la fase de entrenamiento, se probaron de manera manual. Únicamente 7 dieron resultado incorrecto.

Modo	Zona	Balon	BalonZ	BalonMarco	AmigoBalon	ContrarioBalon	Contrario	Esperar	Girar	IrABalon	Cobertura	Desmarcarse	Apoyar	VolverZona	Res
0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	paso
0	0	1	0	0	1	1	1	0	0	0	0	0	0	1	paso
0	0	1	0	1	1	1	0	0	0	0	0	0	1	0	paso
0	0	1	1	0	0	0	1	0	0	1	0	0	0	0	paso
0	0	1	1	1	0	0	0	0	0	1	0	0	0	0	paso
0	0	1	1	1	0	0	1	0	0	1	0	0	0	0	paso
0	1	0	0	0	0	0	1	0	1	0	0	0	0	0	paso
0	1	1	0	0	0	0	1	1	0	0	0	0	0	0	fallo
0	1	1	0	0	0	1	1	1	0	0	0	0	0	0	fallo
0	1	1	0	0	1	1	1	1	0	0	0	0	0	0	fallo
0	1	1	0	1	0	0	1	0	0	1	0	0	0	0	paso
0	1	1	0	1	1	1	1	0	0	0	0	0	1	0	paso
0	1	1	1	0	0	0	0	0	0	1	0	0	0	0	paso
0	1	1	1	1	0	0	0	0	0	1	0	0	0	0	paso
0	1	1	1	1	0	0	1	0	0	1	0	0	0	0	paso
0	1	1	1	1	0	1	1	0	0	1	0	0	0	0	paso
0	1	1	1	1	1	0	0	0	0	1	0	0	0	0	fallo
1	0	0	0	0	0	0	1	0	1	0	0	0	0	0	paso
1	0	1	0	0	1	0	1	0	0	0	0	1	0	0	fallo
1	0	1	1	0	0	0	1	0	0	1	0	0	0	0	paso
1	0	1	1	0	1	0	0	1	0	0	0	0	0	0	paso
1	0	1	1	0	1	1	0	0	0	1	0	0	0	0	paso
1	0	1	1	1	0	1	0	0	0	1	0	0	0	0	paso
1	0	1	1	1	1	0	0	1	0	0	0	0	0	0	paso
1	1	1	0	0	0	1	1	1	0	0	0	0	0	0	fallo
1	1	1	0	1	1	0	1	0	0	0	0	1	0	0	paso
1	1	1	0	1	1	1	0	1	0	0	0	0	0	0	paso
1	1	1	1	0	0	0	1	0	0	1	0	0	0	0	paso
1	1	1	1	0	1	1	0	0	0	1	0	0	0	0	fallo
1	1	1	1	0	1	1	1	0	0	1	0	0	0	0	paso
1	1	1	1	1	0	0	1	0	0	1	0	0	0	0	paso
1	1	1	1	1	0	1	0	0	0	1	0	0	0	0	paso

13 . Referencias Bibliográficas

[ABRA01]

Abraham D., Applying Neural Networks to Soccer, 2001.

[BENA13]

Benarroch G., Entrenadores de Fútbol, Entrenador Nacional de Fútbol, Barcelona España, consultado en Enero 2013 de http://entrenadordefutbol.blogia.com.

[CAGR91]

Carpenter G., Grossberg S., Rosen D., Fuzzy ART: Fast Stable Learning and Categorization of Analog Patterns by an Adaptive Resonance System, Neural Networks, Vol. 4, 1991, p.p 759 – 751, Boston University.

[CARP03]

Carpenter Gail A., ART Neural Networks: Distributed Coding and ARTMAP Applications, Department of Cognitive and Neural Systems, Boston University, Massachusetts, 2003.

[CAST00]

Castro J., Sistemas de Redes Neuronales: Una variante del Algoritmo de Retropropagación NPROP, Tesis para optar al grado de Magister en Computación con énfasis en Scientae en Computación, 2000.

[CAZO99]

Cazorla Q., Colomina O., Escolano F., Gallardo D., Rizo R., Satorre R., Técnicas de Inteligencia Artificial, Publicaciones Universidad de Alicante, 1999,

[CHFO03]

Chen M, Dorer K, Foroughi E, Heintz F, Huang Z, Kapetanakis S, Kostiadis

K, Kummeneje J, Murray J, Noda I, Obst O, Riley P, Steffens T, Wang Y, Yin X, Robocup Soccer Server – User Manual, 2003.

[DRC13]

DARPA, DARPA Robotics Challenge, consultado el 7 de abril 2013, de http://www.darpa.mil/Our_Work/TTO/Programs/DARPA_Robotics_Challenge.aspx.

[DLIN92]

Lin D-T., Dayhoff J.E., Ligomenides P.A., A Learning Algorithm for Adaptive Time-Delays in a Temporal Neural Network, Institute for Systems Research,1992.

[ECOI11]

EcoInteligencia, DARPA Grand Challenge, el desafío para coches sin conductor., consultado el 7 de abril de 2013, de http://www.ecointeligencia.com/2011/11/darpa-grand-challenge-coches-sin-conductor/.

[ELMA90]

Elman Jeffrey, Finding Structure in Time, University of California, San Diego, Cognitive Science, 14, 1990.

[FRGR96]

Franklin S., Graesser A., Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents, Institute for Intelligent Systems, University of Memphis, 1996.

[GARMAY]

García-Herrera R., Mayol-Cuevas W.W., Reconocimiento de Imágenes Empleando Redes de Regresión General y la Técnica TVS, Universidad Nacional Autónoma de México.

[GARC05]

García P., Las redes neuronales supervisadas, consultado en mayo de 2013 en www.telefonica.net/web2/pgestevez/publicaciones.htm, 2005.

[GERS98]

Gerstner W., Supervised learning for neural networks: A tutorial with Java Exercises, consultado en http://diwww. epfl. ch/mantra/tutorial/english/supervised. pdf.

[GIPU13]

Táctica y Estrategia, GipuzkoaKirolak, Diputación Foral de Gipuzkoa, Dirección General del Deporte, consultado en marzo de 2013 de http://www.kirolzerbitzua.net/informacion/webfutbol/pdfs/Futbol%20Cast1-4.pdf

[GIRÁ99]

Giráldez J. I., Modelo de Toma de Decisiones y Aprendizaje en sistemas multi-agente, Tesis Doctoral, Departamento de Inteligencia Artificial, Facultad de Informática, Universidad Politécnica de Madrid.

[GORN97]

Gorni Antonio Augusto, The Application of Neural Networks in the Modeling of Plate Rolling Processes, 1997, Consultado el 20 de marzo de 2012 en **http://www.tms.org/pubs/journals/JOM/9704/Gorni/**.

[GRAN07]

Tácticas de Fútbol, Granfutbol.com, consultado en marzo 2013 en http://granfutbol.com/tacticas.html, 2007.

[GRIF05]

Griffiths N., Cooperative clans, Kybernetes, 2005, 34: 9 – 10.

[GUTI01]

Gutiérrez M., Administración de carteras con redes neuronales mediante metodología Rolling, Seminario para optar al título de ingeniero comercial, Escuela de Economía y Administración, Universidad de Chile, 2001.

[HELO96]

Sistema de Redes Neuronales: Una Modificación al Algoritmo de Aprendizaje de Retropropagación, Tesis para optar al grado de Magister Scientae en Computación con énfasis en Ciencias de la computación, 1996.

[HINT06]

Hinton G., To Recognize Shapes, First Learn to Generate Images, Department of Computer Science, University of Toronto & Canadian Institute for Advanced Research, 2006.

[HINT07]

Hinton G., Learning multiple layers of representation, Department of Computer Science, University of Toronto, 2007.

[HISA06]

Hinton G., Salakhutdinov R. R., Reducing the Dimensionality of Data with Neural Networks, Science, 2006.

[HOFR13]

Hof R., Aprendizaje Profundo, MIT Technology Review, 2013.

[HOKA05]

Hossein D. M., Kaviani N., Nikanjam A., Rafaie-Jokandan M., Training a Simulated Soccer Agent how to Shoot using Artifial Neural Networks, RoboSina Simulation Research Group, Department of Computer Engineering, Bu-Ali Sina

University, Hamedan, Iran, 2005.

[JAMA96]

Jain A., Mao J, Artificial Neural Networks: A Tutorial, 1996.

[JAJA92]

Jájá J., An Introduction to Parallel Algorithms, Addison-Wesley, 1992.

[JUBO00]

V. Julián, V. Botti, Agentes Inteligentes: el siguiente paso en la Inteligencia Artificial, Dpto. Sistemas Informáticos y Computación, Universidad Politécnica de Valencia, 2000.

[KAHU02]

Kalyviotis N., Hu H., A Neural Network decision module for the Adaptive Strategic Planning Framework in RoboCup, Department of Computer Science, University of Essex, 2002.

[KIAS97]

Kitano H., Asada M., Kuniyoshi Y., Noda I., Osawa E., Matsubara H., Robocup A Challenge Problem for AI., AI Magazine, Vol 18, Num 1, 1997.

[LLHO07]

Llano L., Hoyos A., Arias F., Velásquez J., Comparación del Desempeño Funciones de Activación en Redes Feedforward para aproximar Funciones con Datos con y Sin Ruido, Interconexión Eléctrica S.A. E.S.P. (ISA), Colombia, GIDIA: Grupo de Investigación y Desarrollo en Inteligencia Artificial, Grupo de Finanzas Computacionales, Escuela de Ingeniería de Sistemas, Facultad de Minas, Universidad Nacional de Colombia Sede Medellín, 2007.

[MERE13]

Merelo J.J., Mapa autoorganizativo de Kohonen, consultado en 8 de mayo de 2013 en http://geneura.ugr.es/~jmerelo/tutoriales/bioinfo/.

[MIRA00]

Miraftabi R., Agents on the Loose: An Overview of Agent Technologies, Department of Computer Science, University of Joensuu, 2000.

[OTSA05]

Ottair Mohammed A., Salameh Walid A., Speeding Up Back-Propagation Neural Networks, Proceedings of the 2005 Informing Science and IT Education Joint Conference, 2005.

[PAGO02]

Parrado E., Gómez E., Dimitriadis Y., Study of distributed learning as a solution to category proliferation in Fuzzy ARTMAP based neural systems, Neural Networks 16 (2003), 2002, p.p 1039-1057.

[PETR96]

Petrie Charles, Agent-Based Engineering, the Web, and Intelligence., 1996.

[ROBO13]

The Robocup Federation, Robocup web site, consultado en www.robocup.org.

[ROCA03]

Roca C., Agentes Informáticos: Deciden por sí solos y reaccionan a los cambios, Automática e Instrumentación, 2003, 340: 74 – 77.

[ROJA96]

Rojas R., Neural Networks, Springer-Verlag, Berlin, 1996.

[RUWI94]

Rumelhart D, Widrow B, Lehr M, The Basic Ideas in Neural Networks., 1994.

[RUNO10]

Russel S., Norvig P., Artificial Intelligence A Modern Approach. Third Edition., Prentice Hall, 2010.

[RUSS05]

Russel I., Neural Networks, Department of Computer Science, University of Hartford, 2005,

[SEDE04]

Serenko, A., Detlor B., Intelligent agentes as innovations, AI & Soc, 2004, 18: 364 – 381.

[STAR00]

Starikov Alexey, Self-Organizing Maps – Mathematical Apparatus, BaseGroup Labs, 2000.

[TODD91]

Tood Petter M., Neural Networks for Applications in the Arts, Department of Psychology, Sanford University, California, 1991.

[WEEN97]

Weenink David, Category ART: A variant on Adaptive Resonance Theory Neural Networks, Institute of Phonetic Sciences, University of Amsterdam, (Proceedings 21) 1997.

[WOJE95]

Wooldridge, M., Jennings N., Intelligent agents theory and practice, Knowledge Engineering Review, 1995.

[ZABO05]

Zarour N., Boufaida M., Seinturier L., Estraillier P., Supporting virtual enterprise systems using agent coordination, Knowledge and Information Systems, 2005, 8: 330 – 349.