CARRERA DE ESPECIALIZACIÓN EN INTELIGENCIA ARTIFICIAL

MEMORIA DEL TRABAJO FINAL

Módulo de procesamiento de lenguaje natural para el proceso de enseñanza y aprendizaje en una sociedad de robots

Autor:
Ing. Bryan Leandro Quiroga Gavilán

Director:
MSc. Ing. Dante Sterpin Buitrago (CUN)

Jurados:
Esp. Ing. Leandro Bello (FIUBA)
Esp. Ing. Maximiliano Torti (FIUBA)
Esp. Dis. Ana Guzmán (FIUBA)

Este trabajo fue realizado en la Ciudad de Bogotá, Colombia, entre noviembre de 2022 y agosto de 2023.

Resumen

La presente memoria describe el diseño e implementación de un generador de lenguaje natural que permite a un robot instructor transformar un conjunto de valores numéricos en expresiones lingüísticas relacionadas con la actividad de seguimiento de pared, con el objetivo de facilitar un proceso de enseñanza más natural en la sociedad de robots del sistema de co-evolución genética y neuro-memética (GeNeSys).

Para lograr este propósito, se generó un conjunto de datos utilizando un sistema de inferencia difusa. Además, se consideraron categorías como preprocesamiento de texto, redes neuronales recurrentes y modelos de aprendizaje profundo modernos.

Agradecimientos

A MEd. Lic. Karen Tatiana Ramos Castellanos a quién no puedo agradecerle lo suficiente por el incalculable ánimo y calma que nos permite ser un gran equipo.

A mis hermanas, mis padres y mis mejores amigos de la cúpula por brindarme alegría y apoyo en todo momento.

A mi director el MSc.Ing. Dante Sterpin por apoyar y mostrar total interés en el desarrollo de este trabajo para que haga parte de uno de sus más grandes proyectos.

A los acompañantes del posgrado en la revisión de esta memoria por el tiempo dedicado.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción general

En este capítulo, se detallan los conceptos básicos de la generación de texto a partir de datos, comúnmente conocida como D2T (*Data2Text*) y la lógica difusa, que constituyen la base teórica de este trabajo. Además, se referencian investigaciones efectuadas por otros autores que han realizado tareas que se pueden considerar similares. También se describe el objetivo y las actividades que delimitan el alcance del modelo propuesto de generación de lenguaje.

1.1. Concepto datos a texto

La generación de texto a partir de datos es un campo de investigación en el que se busca producir expresiones lingüísticas que describan adecuada y fluidamente entradas no lingüísticas como tablas de bases de datos, artículos o simulaciones de sistemas físicos [1].

Existen diferentes enfoques principales para generar texto a partir de datos, entre ellos se encuentran el uso de reglas, plantillas y redes neuronales. Los métodos basados en reglas y plantillas son ampliamente utilizados en diversas aplicaciones debido a su capacidad para brindar un control e interpretación claros de los datos, lo que ayuda a garantizar la precisión del texto generado. Si bien estos métodos requieren un trabajo manual intenso para definir características y lograr una alta calidad en las reglas y plantillas, en ocasiones son necesarios para obtener resultados precisos y efectivos.

Los modelos basados en redes neuronales se fundamentan principalmente en datos y requieren menos intervención humana. Estos modelos pueden generar fácilmente textos descriptivos y fluidos, pero resulta más desafiante garantizar que los textos generados sean fieles a los datos de entrada [2].

1.2. Lógica difusa

En 1965, Lotfi A. Zadeh introdujo la lógica difusa definida como una forma de lógica multivaluada que permite definir valores intermedios entre las evaluaciones tradicionales, como verdadero/falso, sí/no, alto/bajo, etc. Esto facilita la aplicación de un enfoque más natural e intuitivo en la programación de computadoras al permitir la formulación matemática y el procesamiento de terminología ambigüa, tal como “bastante alto” o “muy rápido” [3].

Considerando un conglomerado de elementos discretos o continuos, un conjunto difuso es una función que define matemáticamente el grado con el que cada elemento pertenece a cierta categoría [4]. Así, cada elemento es clasificado mediante un valor comprendido entre 0 y 1. Estos conjuntos difusos formalizan numéricamente la ambigüedad inherente en dicha clasificación, ampliando así la noción de inclusión y permitiendo operaciones tales como la unión, intersección y complemento entre clases de elementos.

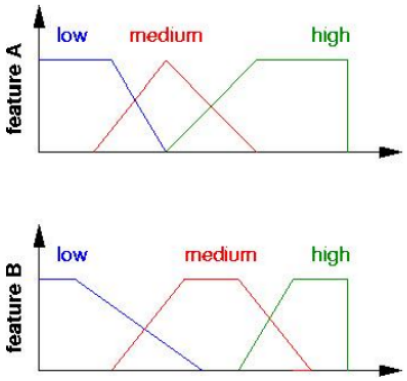


FIGURA 1.1. Ejemplo de conjuntos difusos de 2 características [1].

1.3. Planteamiento y oportunidad del problema

Implementado por [5], el sistema de co-evolución genética y neuro-memética (GeNeSys) consiste en una sociedad de robots capaces de desarrollar y propagar culturalmente cierto comportamiento, mediante la interacción social entre "instructores" e "imitadores", usando un modelo computacional de replicación neuro-memética que fue validado mediante sistemas conexionistas tales como el mapa auto-organizado (SOM, del inglés, *Self-Organizing Maps*) y la red adaptativa de inferencia difusa (ANFIS, del inglés, *Adaptive-Network based Fuzzy Inference System*).

En el respectivo proceso cultural de enseñanza-aprendizaje, cada robot entrega valores numéricos que, por un lado, representan el comportamiento que es exhibido por algún robot, y por otro lado, representan la información que este puede compartir con otros robots. La vinculación de un generador de lenguaje natural (NLG, del ingles *Natural Language Generation*) al sistema GeNeSys busca satisfacer una necesidad previamente identificada por el líder del proyecto, en cuanto a incrementar la naturalidad en la interacción entre los individuos bio-culturales, allí denominados: GeNeBots. Eventualmente, esta propuesta investigativa puede servir de base para que en diversos ámbitos empresariales puedan desarrollarse modelos que interpreten lingüísticamente datos numéricos, expresándose en la terminología propia del contexto en donde se pretenda aplicar, ya sea bancario, médico, de manufactura, u otros.

[1] Imagen tomada de [3]

1.4. Estado del arte

La generación de texto a partir de datos es un campo en constante evolución y desarrollo. En este existen diversas herramientas y técnicas disponibles para generar texto automáticamente desde datos estructurados o no estructurados, así como herramientas en línea que utilizan inteligencia artificial y algoritmos complejos basados en reglas. Algunas utilizan técnicas avanzadas de procesamiento del lenguaje natural donde la calidad del texto generado depende en gran medida de la calidad y cantidad de datos disponibles; así como de la capacidad de la herramienta para analizar y comprender los datos. En esta sección se describirán algunas investigaciones previas que han servido de guía para la elaboración de este trabajo.

1.4.1. Generación neuronal consciente de la lógica

Esta investigación presenta un método que se enfoca en detectar posibles riesgos en los datos de entrada y explicar en lenguaje natural comportamientos anormales correspondientes al lavado de dinero. El enfoque propuesto, llamado generación neuronal consciente de la lógica (LANG, del inglés *Logic Aware Neural Generation*), combina la modelización lógica con la generación de texto. Para lograrlo, se utilizan reglas expertas que se convierten en un grafo lógico, y un codificador basado en *metapaths* que aprovecha el conocimiento especializado [6]. Además, se emplea un módulo recuperador basado en capas de atención para vincular los datos numéricos de entrada con el texto objetivo y una estrategia de pérdida basada en las reglas mencionadas que mejora la precisión en la generación de texto (ver figura 1.2)

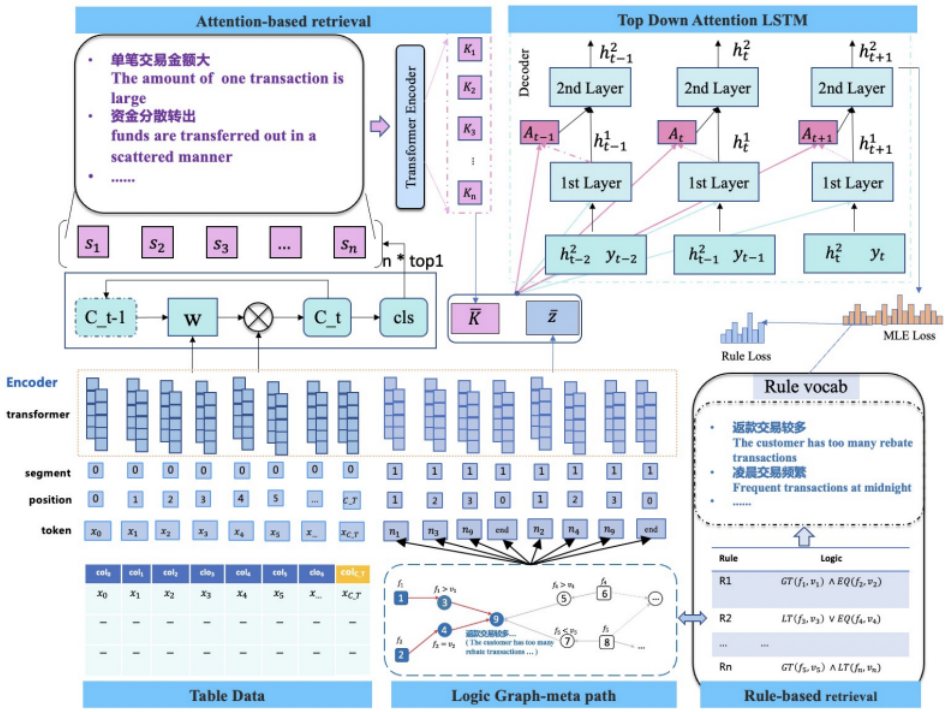


FIGURA 1.2. Arquitectura de LANG [2].

[2] imagen tomada de [6]

1.4.2. Índice de calidad del aire

En este estudio, se presenta un enfoque basado en inteligencia computacional y generación de lenguaje natural para la elaboración automática de resúmenes a partir de series de datos numéricos. El objetivo es proporcionar información relevante oculta en los datos y facilitar la comprensión a los usuarios utilizando lo que los autores denominan ontología temporal difusa. El objetivo es realizar una evaluación exhaustiva utilizando datos reales del índice de calidad del aire (ICA) [7].

El proyecto utiliza 38 reglas difusas en forma trapezoidal que componen el diseño gramatical del resumen ICA, y ha recibido calificaciones satisfactorias por parte de los expertos. Actualmente el modelo se encuentra en funcionamiento en la web oficial de MeteoGalicia [7].

1.4.3. Desafío en la generación de texto a partir de datos

Challenge in Data-2-text Generation es un trabajo que plantea una tarea más desafiante de generación de texto a partir de datos y se presenta un nuevo corpus a gran escala que contiene registros de datos emparejados con documentos descriptivos. Los experimentos muestran que dichos modelos generan texto de forma fluida, pero no logran aproximarse de manera convincente a la calidad de los documentos generados por humanos [8].

El modelo base utilizado realiza un *embedding* de algunas características principales de forma individual insertándolas en un perceptrón multicapa (MLP, del inglés *Multi-Layer Perceptron*) de una capa oculta para pasar a un modelo neuronal recurrente tipo LSTM, del inglés *Long-Short Term Memory*.

1.4.4. Operaciones guiadas por modelos neuronales secuencia a secuencia *seq2seq*

Esta propuesta denominada OpAtt (abreviatura del inglés *Operation guided Attention based sequence to sequence network*) consiste en utilizar información de operaciones pre-ejecutadas en los datos de entrada para guiar la generación de texto. El modelo propuesto consta de un codificador de registros, un codificador de operaciones y un codificador de resultados de operaciones, junto con un decodificador de unidad recurrente con compuertas (GRU, del inglés *Gated Recurrent Unit*) equipado con atención y un mecanismo de compuerta (ver figura 1.3). Los resultados pre-ejecutados obtenidos de las operaciones actúan como hechos inferidos para guiar la generación y se utiliza una capa de cuantización para establecer correspondencias entre resultados numéricos y elecciones léxicas [9].

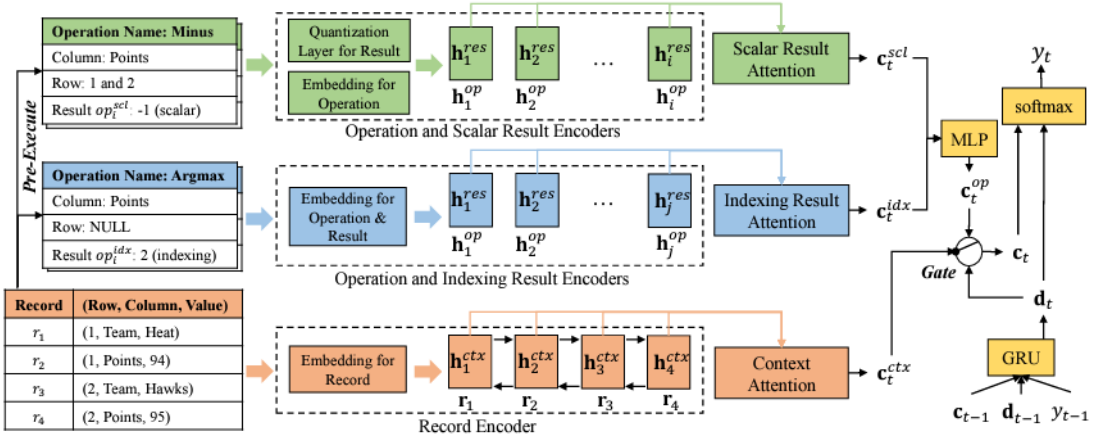
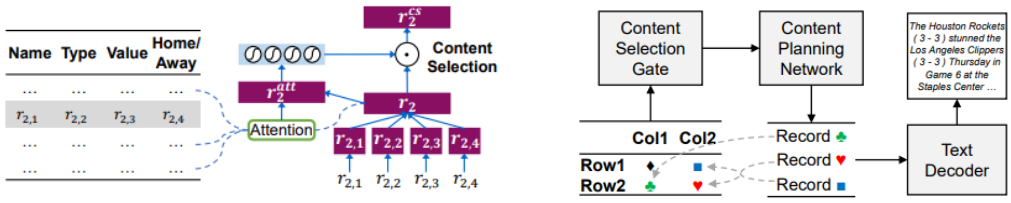


FIGURA 1.3. Pre-ejecución de datos y arquitectura de OpAtt [3].

1.4.5. Selección de contenido, macroplanificación y planificación secuencial variacional

Las investigaciones lideradas por Ratish Puduppully se centran en el enfoque D2T utilizando conjuntos de datos deportivos y métodos de extracción de información para ser incorporados en el modelo base mencionado en la sección 1.4.3. A continuación, se describen brevemente estos trabajos y los cambios que se han realizado entre ellos [10] [1].

- Selección de contenido y planeación: una capa adicional denominada *content select gate* tiene la finalidad de calcular puntuaciones para cada entrada en relación con los demás datos del conjunto tabular como se evidencia en la figura 1.4 a. Estas puntuaciones se someten a una función sigmoide y se realiza una operación de producto punto con los valores de entrada y así genera una matriz que representa el selector de contenido [10]. Esta mejora en el modelo base permite resaltar la relevancia de ciertos datos durante el proceso de generación de texto para llegar a la etapa de decodificación del texto (ver figura 1.4 b).



(a) (b)

FIGURA 1.4. a) Esquema de selector de contenido. b) Flujo de procesamiento del modelo [4].

[3] Imagen tomada de [9]

[4] Imagen tomada de [10]

- Macroplanificación: es una ampliación del modelo de selección de contenido y planificación que involucra un preprocesamiento de datos a un nivel más alto. En este contexto, se define un macro plan como una secuencia estructural de párrafos que se separan mediante el uso de una etiqueta o marcador de discurso <P>. Cada plan de párrafo describe una secuencia de entidades y eventos que se refieren a jugadores individuales o equipos y su desempeño en un juego de béisbol. Las entidades y eventos se verbalizan en una secuencia de texto utilizando tokens especiales para indicar el tipo y valor de cada registro en la tabla de datos.
- Planificación secuencial variacional: el enfoque utiliza técnicas de inferencia variacional para modelar la distribución latente de los macroplanes en la generación de texto a partir de datos estructurados. El modelo consiste en un codificador que mapea los datos de entrada a una distribución latente y un decodificador que genera los macroplanes. Durante el entrenamiento, se maximiza la evidencia inferior para capturar la incertidumbre inherente en la generación de planes secuenciales [1]. Esto permite al modelo generar macroplanes de alta calidad y capturar la coherencia y estructura del texto final.

1.4.6. Tabla comparativa de modelos D2T

La comparativa realizada de los modelos se hace en referencia a la aplicabilidad al modelo NLG del sistema GeneSys; el nivel de aplicabilidad se define con base al nivel de preprocesamiento de datos y complejidad del modelo, motivos por los que puede ser ineficiente computacionalmente en un eventual despliegue.

TABLA 1.1. Comparación de modelos en generación de texto.

Modelo	Preprocesado	Complejidad	Aplicabilidad
LANG	Media	Alta	Media
Ontología temporal difusa	Media	Media	Media
Operaciones guiadas OpAtt	Alto	Alta	Media
Selección de contenido	Muy Alto	Media	Limitada
Macroplanificación	Muy Alto	Media	Limitada
Planificación secuencial variacional	Muy alto	Alta	Limitada

1.5. Motivación

La literatura existente revela que los modelos actuales para esta tarea son complejos y requieren un extenso preprocesamiento de datos, esto limita su aplicabilidad en situaciones prácticas. La motivación detrás de este proyecto es desarrollar un enfoque más sencillo y efectivo que permita generar instrucciones lingüísticas en el proceso de enseñanza-aprendizaje del proyecto GeNeSys, con la intención de lograr un modelo sencillo que facilite la generación de texto en diversos contextos. Así, este trabajo pretende simplificar el proceso D2T y obtener resultados de calidad sin la necesidad de técnicas complejas de preprocesamiento.

1.6. Objetivos y alcance

Considerando los 4 valores numéricos (P, D, L, R) que definen el comportamiento de los robots en el sistema GeNeSys, el objetivo de este proyecto es desarrollar un modelo generador de lenguaje natural, que sea capaz de producir expresiones lingüísticas basadas en dichos valores. Esto le permitiría a un robot "instructor" verbalizar su propio comportamiento, con el fin de recomendárselo a otros robots. Para lograr lo propuesto se plantean los siguientes objetivos específicos:

- Obtener una base de datos, a partir de un sistema de inferencia difusa (FIS, del inglés *Fuzzy Inference System*), que sea útil para entrenar el modelo propuesto.
- Realizar un preprocesamiento de datos corto que permita al modelo un entrenamiento rápido y una generación de secuencias que sean computacionalmente eficientes.
- Implementar el modelo propuesto en ambiente de simulación del sistema GeNeSys.
- Evidenciar las secuencias generadas con datos de prueba provenientes del FIS y los generados de los sistemas conexionistas del GeNeSys.

Capítulo 2

Introducción específica

En este capítulo se detallan algunos de los conceptos fundamentales en el uso de lenguaje natural, por parte de los sistemas de inferencia difusa y en la generación de lenguaje natural (NLG) por parte de las redes neuronales recurrentes. Así mismo, se detallan algunos aspectos del sistema GeNeSys que son claves para la generación de las variables de entrada.

2.1. Sistema de inferencia difusa tipo Mamdani

Propuesto en 1975 por Ebrahim Mamdani, es probablemente el método más utilizado como mecanismo de inferencia difusa [11]. Es un enfoque utilizado para la construcción de sistemas de control difuso que se fundamenta en reglas difusas si-entonces (*IF-THEN*) y utiliza el concepto de conjuntos difusos mencionados en la sección 1.2 para representar la incertidumbre y la imprecisión en los datos de entrada y salida.

El FIS Mamdani es práctico y eficiente para definir términos lingüísticos y conjuntos difusos. por lo tanto, permite generar un conjunto de datos que se aproxime a las características de los sistemas conexionistas de los robots del GeNeSys, que dependen de la lógica difusa (ver sección 2.2.1). Para lograr esto, el FIS Mamdani sigue una serie de pasos: fuzzyficación, activación de reglas difusas y defuzzyficación. A continuación, se describen estos pasos.

2.1.1. Fuzzyficación

En esta fase, se calcula el grado de pertenencia de esas variables a todos los conjuntos difusos posibles [12]. En la figura 2.1 se puede observar el valor de membresía para dos conjuntos difusos denominados “baja” y “media” para un determinado valor de temperatura.

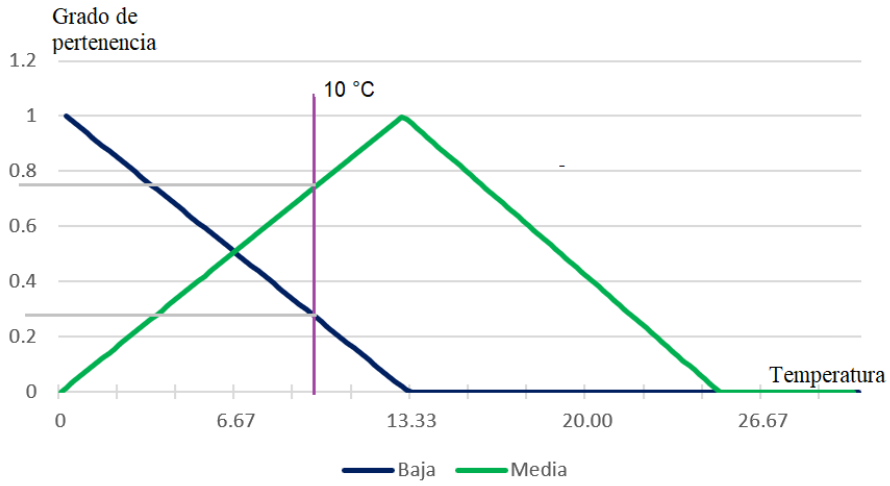


FIGURA 2.1. Función de membresía temperatura baja y media para 10 °C[1].

Para definir un conjunto difuso triangular, se establecen los límites a (límite inferior), b (límite superior) y m (punta del triángulo). El valor de pertenencia se calcula de acuerdo con la ecuación 2.1.

$$\mu(x) = \begin{cases} 0 & \text{si } x \leq a \\ \frac{x-a}{m-a} & \text{si } a < x \leq m \\ \frac{b-x}{b-m} & \text{si } m < x < b \\ 0 & \text{si } x \geq b \end{cases} \tag{2.1}$$

2.1.2. Activación de las reglas difusas

Los modelos difusos de Mamdani no necesitan modelos matemáticos del sistema a controlar y se obtienen a partir de reglas difusas o enunciados condicionales difusos. Por ejemplo, si el error de presión es negativo y grande, entonces el cambio de calor es positivo y grande [13], donde el término "error" se refiere a la diferencia entre el valor real de la variable y el punto de referencia (*setpoint*). En general, las reglas de tipo Mamdani tienen la siguiente forma:

$$R_i : if X is A_i, X is A_i, ..., X is A_i then Y is B_i. \tag{2.2}$$

Aquí es donde se deben realizar las operaciones con los conjuntos difusos predeterminados. Existe una generalización de las funciones que definen la unión y la intersección de conjuntos difusos, conocidas como conorma triangular (T-Conorma) y norma triangular (T-Norma). En la mayoría de las aplicaciones de ingeniería basadas en lógica difusa, se opta por utilizar el operador máximo como T-Conorma y el operador mínimo como T-Norma [14], el último utilizado en los propósitos del FIS de este trabajo. La intersección de dos conjuntos difusos A y B es un conjunto difuso A ∩ B en U con función de pertenencia definida en la ecuación 2.3.

[1]Elaboración propia basada en [12]

$$\mu_{A\cap B}(x) = \text{mín}[\mu_A(x), \mu_B(x)] \tag{2.3}$$

2.1.3. Defuzzyficación

Es el proceso de tomar las salidas difusas y convertirlas en un valor de salida único o nítido. Este proceso puede ser realizado por cualquiera de varios métodos de defuzzificación [15]. En esta investigación se utilizó el método de centroide calculado con la ecuación 2.4

$$Centroide = \frac{\sum_{i=1}^{n}(Y_i \times \mu_i)}{\sum_{i=1}^{n} \mu_i} \tag{2.4}$$

2.2. Sistema GeNeSys

El sistema de co-evolución genética y neuro-memética (GeNeSys) es un modelo computacional de la emergencia y propagación cultural de cierto comportamiento, en una sociedad de robots. Allí cada robot, como individuo autónomo, está dotado con sistemas conexionistas muy sencillos capaces de aprendizaje autónomo y social, cuya topología es heredada mediante mecanismos genéticos. Con dichos sistemas, cada robot puede descubrir por sí mismo patrones comportamentales útiles, o bien "imitar" a otros individuos que ya tengan el conocimiento. En el escenario "imitativo", es en donde se utiliza la replicación neuro-memética como mecanismo de herencia cultural [16] y por el efecto Baldwin, los individuos con mejores comportamientos adquiridos y mejor habilidad para "enseñarlos" son los más atractivos en la reproducción sexual. En términos de robótica evolutiva, dicha sociedad modela la herencia dual de genes y neuro-memes [16] para explorar y explotar controladores adaptativos; su objetivo es seguir explorando y explotando estos controladores hasta que uno de los individuos logre realizar exitosamente una tarea específica y demuestre su eficacia también en tareas colaborativas.

En contraste con otros modelos sociales y evolutivos, el GeNeSys es un sistema multi-agente completamente descentralizado y se enfoca en el desarrollo cultural mediante replicadores que no viajan entre sus huéspedes, razón por la que deben transmitir su información mediante señales sociales, exclusivamente.

2.2.1. Sistema de inferencia difusa basada en red adaptativa

La red ANFIS es un sistema de inferencia difusa que posee la capacidad de aprender a relacionar las entradas y salidas a través del aprendizaje supervisado. A diferencia de una red neuronal convencional, ANFIS es una red adaptativa en la que no existen pesos de conexión, así los parámetros que se aprenden están asociados a nodos específicos.

En la capa 1 de ANFIS, se lleva a cabo un proceso de fuzzyficación similar al mencionado en la sección 2.1.1 para capturar la incertidumbre y la imprecisión asociadas a los datos. En la capa 2 se estimulan las reglas difusas con AND difusa, en la capa 3 se realiza la normalización de los datos y por último en la capa 4, se emplea una decisión inferencial tipo Sugeno. En la figura 2.2 se puede identificar la arquitectura del ANFIS del GeNeSys.

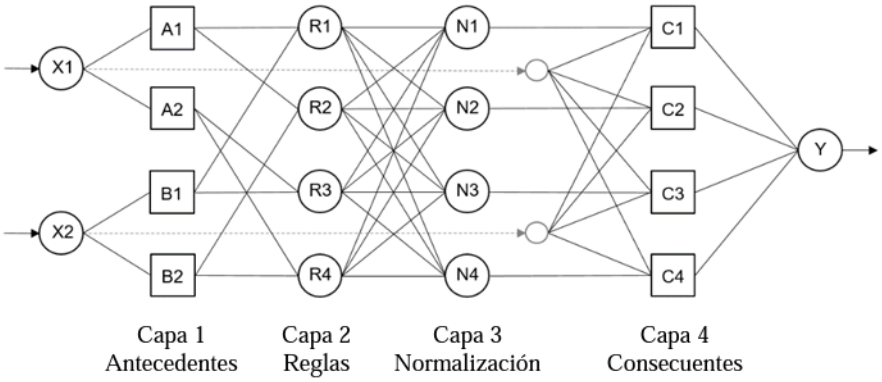


FIGURA 2.2. ANFIS del sistema GeNeSys [2].

2.2.2. Mapa auto organizado

Se define un modelo que busca representar un espacio multidimensional de entrada en un espacio de salida de menor dimensión. La capa de entrada recoge y canaliza la información, mientras que la capa de procesamiento realiza una proyección conservando las características esenciales de los datos mediante una relación de vecindad entre las neuronas [17]. Al final, se obtiene un mapa autoorganizado que muestra agrupaciones en el espacio de salida, lo que facilita la visualización y comprensión de patrones y relaciones entre los datos.

2.3. Variables de entrada

Los datos *clusterizados* del SOM son aquellos P, D, L y R seleccionados para ser compartidos por el potencial robot instructor en el proceso de enseñanza. A continuación se realiza una breve descripción de cada uno de los valores.

1. Proporcional (P): los robots del GeNeSys están dotados con sensores de distancia que se ponderan de manera que el robot tenga una sensibilidad basada en lóbulos gaussianos [5]. Además, el robot está equipado con sensibilidad al choque contra la pared, que en conjunto resulta en valores continuos dentro del rango de -1 a 1 y forma parte del universo de discurso para esta variable, donde -1 representa la posición ponderada más alejada de la pared y 1 representa la posición más cercana a la pared.
2. Derivativo (D): esta variable representa la pendiente resultante de la diferencia entre el valor P de la muestra sensorial actual y el valor P de la muestra inmediatamente anterior. También se encuentra en un rango de -1 a 1, siendo -1 una relación de cambio no pronunciada hacia la pared y 1 una diferencia muy pronunciada que es una aceleración máxima de acercamiento a la pared.
3. Motores izquierdo y derecho (L, R): son los valores que representan la velocidad angular de los motores. Al igual que los anteriores valores, se encuentran en un rango continuo de -1 a 1, resultantes de la decisión del ANFIS y la clusterización del SOM.

[2] Imagen tomada de [5]

2.4. Arquitecturas con redes neuronales recurrentes

En esta sección, se detallan dos arquitecturas clave en el campo de la generación D2T: LSTM presentada por primera vez en [18] y unidad recurrente de compuertas (GRU) igualmente presentada en [19]. Estas arquitecturas, basadas en redes neuronales recurrentes (RNN, del inglés *Recurrent Neural Network*), han demostrado su eficacia en la generación de secuencias de texto coherentes y relevantes. Además, se examinará cómo estas arquitecturas forman parte del famoso modelo Seq2Seq (*sequence-to-sequence*), utilizado ampliamente en tareas de traducción automática y generación de texto.

2.4.1. Celdas de memoria de largo y corto plazo

Según [18], una LSTM es una arquitectura de RNN diseñada para superar el problema del gradiente desvaneciente y capturar dependencias a largo plazo en secuencias de datos.

Como se ilustra en la figura 2.3, en una LSTM se introducen unidades de memoria especializadas llamadas "celdas de memoria" que tienen la capacidad de almacenar y propagar información a lo largo del tiempo. Estas celdas de memoria están compuestas por una unidad central con una conexión autoalimentada fija llamada carrusel de error constante (CEC). Junto con el CEC, se utilizan dos compuertas adicionales para regular el flujo de información: una compuerta de entrada (*input gate*) y una compuerta de salida (*output gate*), donde la compuerta de entrada decide cuánta información nueva se debe agregar a la celda de memoria y la compuerta de salida controla cuánta información de la celda de memoria se debe transmitir hacia afuera. Además, una compuerta de olvido (*forget gate*) determina cuánta información anterior se debe descartar de la celda de memoria.

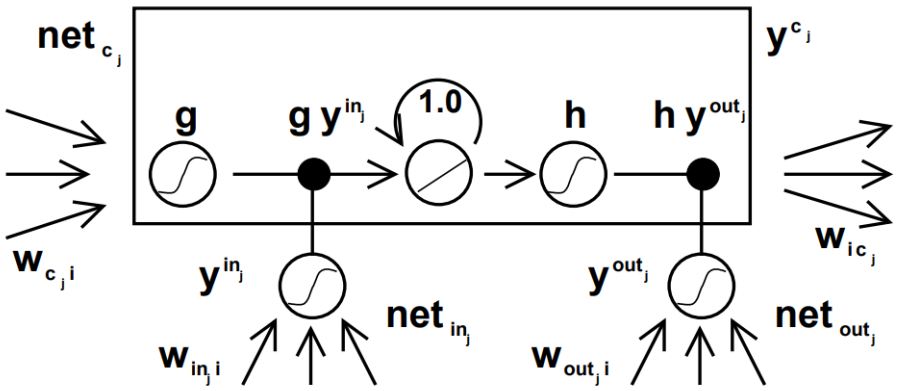


FIGURA 2.3. Arquitectura de LSTM presentado en [18][3].

2.4.2. Unidad recurrente con compuertas

En la investigación realizada por [19] se propone una alternativa a la unidad LSTM, conocida como Unidad Recurrente Cerrada (GRU), con el objetivo de mejorar el rendimiento en tareas de modelado de secuencias.

[3]Imagen tomada y editada de [18]

La GRU se caracteriza por tener dos compuertas principales (ver figura 2.4): una de actualización (*update gate*) y una de reinicio (*reset gate*) que desempeñan un papel fundamental en el control del flujo de información dentro de la unidad. La compuerta de actualización determina qué información del estado anterior se debe mantener y qué nueva información se debe agregar, mientras que la compuerta de reinicio controla qué información del estado anterior se debe olvidar. Así, las compuertas permiten que la GRU capture dependencias a largo plazo al decidir qué información es relevante y debe mantenerse; de igual forma, decide qué información es irrelevante y debe descartarse.

En cada paso de tiempo, la GRU toma el vector de entrada actual $h(k)$ y el estado oculto anterior $h(k-1)$. Luego, se calculan las activaciones de las compuertas de actualización y reinicio utilizando funciones sigmoideas, así estas activaciones se utilizan para calcular el nuevo estado oculto, que es una combinación del estado oculto anterior y el nuevo candidato a estado oculto (*state candidate gate*). Este nuevo estado oculto se pasa al siguiente paso de tiempo y también se utiliza como salida de la unidad.

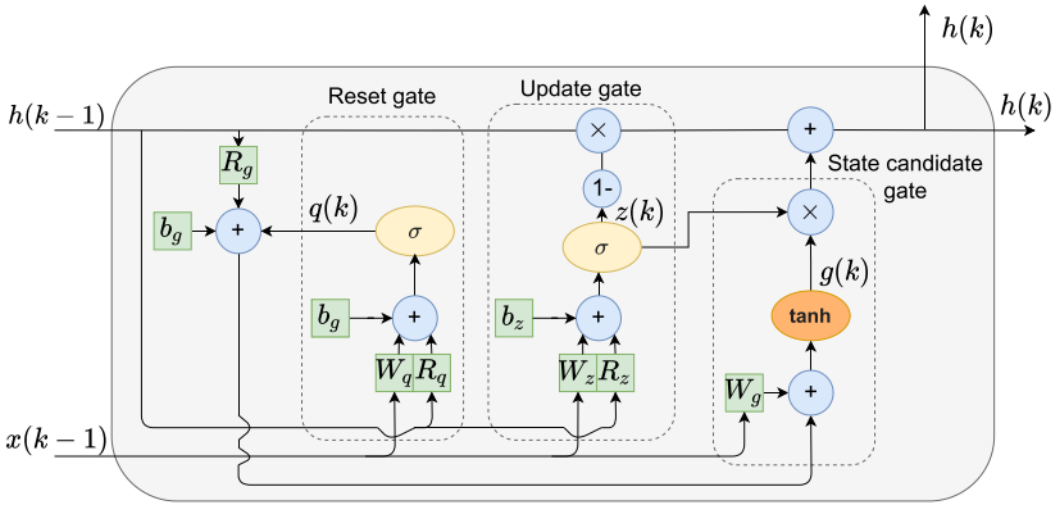


FIGURA 2.4. Arquitectura de GRU [4].

2.4.3. Modelo secuencia a secuencia con RNN

En la investigación efectuada por [21] del equipo de Google, se presenta un enfoque basado en RNN para abordar el desafío de la traducción automática. En lugar de depender de técnicas tradicionales basadas en modelos de lenguaje estadísticos o basados en frases, los autores proponen el uso de un codificador y un decodificador para transformar secuencias de entrada en secuencias de salida correspondientes.

El codificador consiste en celdas LSTM que procesan la secuencia de entrada palabra por palabra y genera una representación vectorial de estado oculto que captura la información relevante de la secuencia. Esta representación de estado oculto condensa la secuencia de entrada en un vector fijo de dimensionalidad fija. El decodificador, que también se compone de celdas LSTM, recibe el estado oculto del

[4] Imagen tomada de [20]

codificador y lo utiliza como contexto para generar una secuencia de salida paso a paso.

El modelo fue entrenado utilizando un corpus paralelo en pares de oraciones correspondientes a los idiomas de origen y destino. Se empleó un enfoque de entrenamiento supervisado en el qué se introduce la secuencia de entrada en el codificador y se entrena el decodificador para generar la secuencia de salida correspondiente. Se llevaron a cabo experimentos comparativos que demostraron que el modelo secuencia a secuencia es considerablemente más eficiente que los métodos basados en modelos de lenguaje estadísticos y en frases. Por ende, sus resultados reflejan mejoras significativas en la calidad de la traducción.

2.4.4. Comparativa de redes neuronales recurrentes

Los resultados de los experimentos revelan que las unidades LSTM y GRU superan a las unidades recurrentes más tradicionales que utilizan la función de activación tangente hiperbólica (tanh). Se ha demostrado que las celdas con mecanismos de compuertas son más efectivas para capturar dependencias a largo plazo en las secuencias y lograr mejores resultados [22]. La figura 2.5 ilustra dichos resultados en dos conjuntos de datos de sonidos polifónicos, considerando diferentes épocas de entrenamiento y tiempo. Aunque la GRU es más eficiente computacionalmente debido a su menor cantidad de parámetros (ver tabla 2.1), sus resultados son similares a los obtenidos por la LSTM.

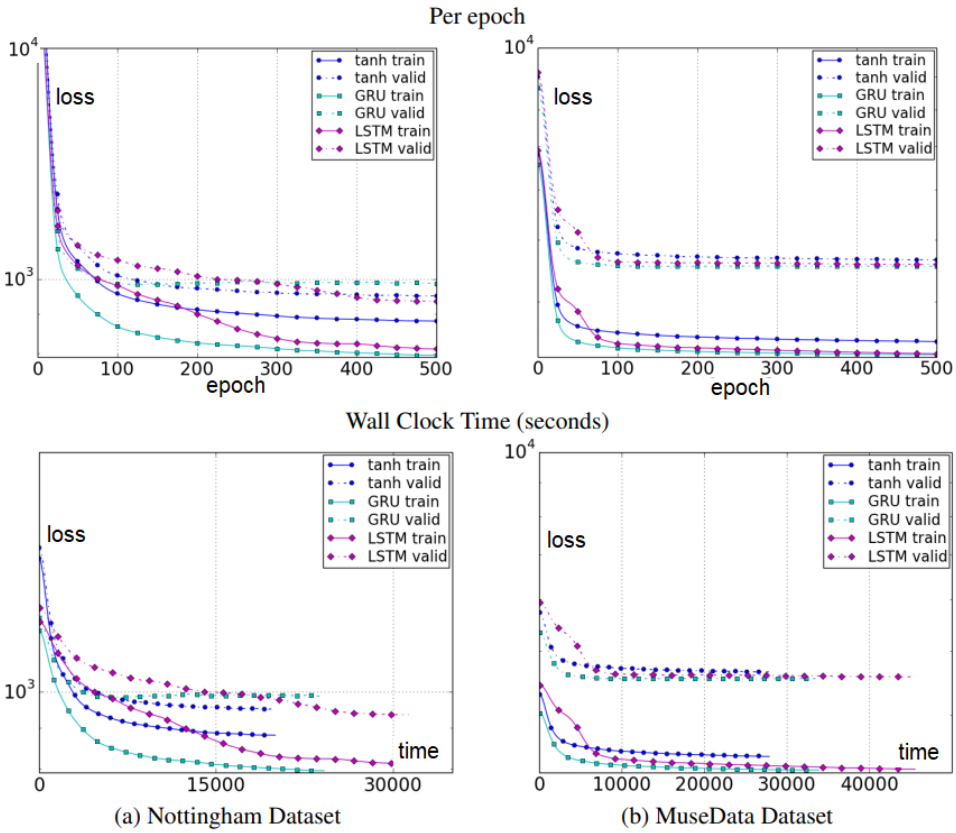


FIGURA 2.5. Resultados de RNNs en *datasets* de sonidos polifónicos [5].

[5]Imagen tomada de [22]

Con los modelos para señales de habla presentados en la tabla 2.1, se muestran los resultados detallados en la figura 2.6. Estos resultados respaldan la afirmación de que tanto GRU como LSTM son comparables en términos de sus métricas de rendimiento en el contexto de señales de habla y que GRU tiene una ventaja significativa en términos de eficiencia computacional.

TABLA 2.1. Modelos utilizados en [22] para realizar pruebas de rendimiento en conjuntos de datos.

Conjunto de datos	RNN	Neuronas	Parámetros
Sonidos polifónicos	tanh	100	≈ 20100
Sonidos polifónicos	LSTM	36	≈ 19800
Sonidos polifónicos	GRU	46	≈ 20200
Señales de habla	tanh	400	≈ 168400
Señales de habla	LSTM	195	≈ 169100
Señales de habla	GRU	227	≈ 168900

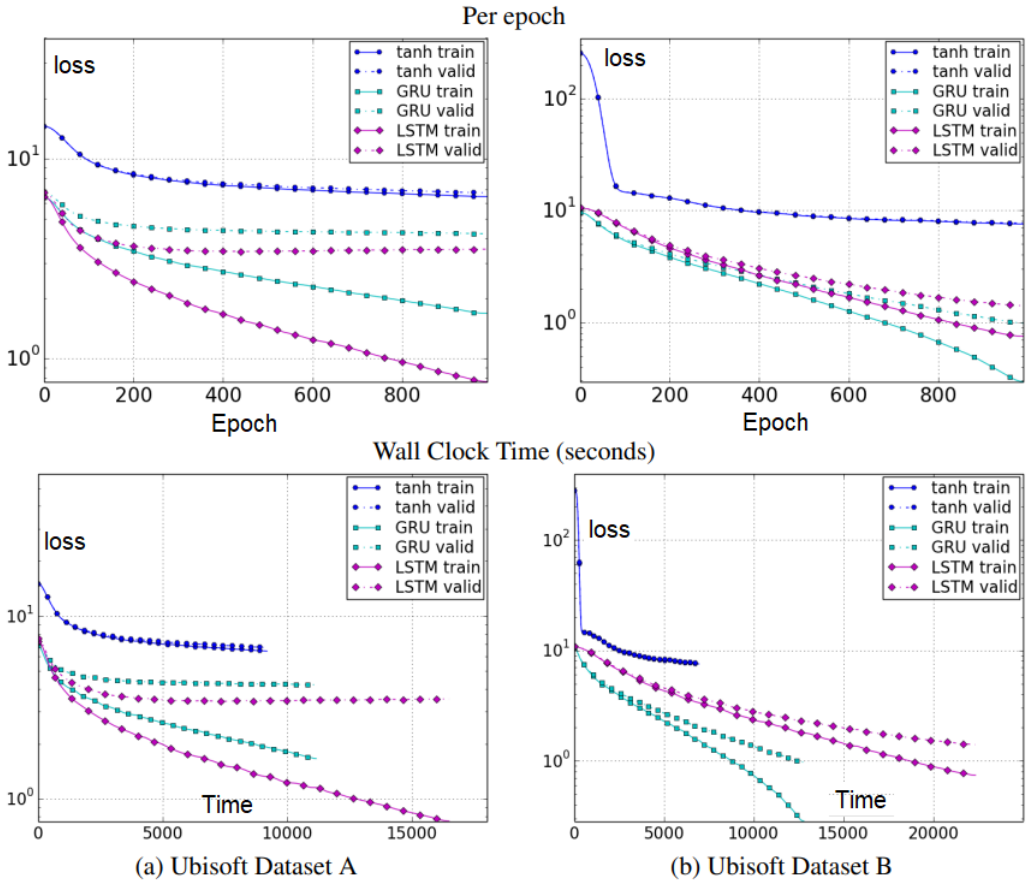


FIGURA 2.6. Resultados de RNNs en *datasets* de señales de habla [6].

[6] Imagen tomada de [22]

Capítulo 3

Diseño e implementación

En esta sección, se abordará la aplicación del FIS Mamdani para generar el dataset utilizado en el entrenamiento del modelo *seq2seq*. Además, se describirá cómo se realizó la clasificación de las frases objetivo y cómo se llevó a cabo su preprocesamiento para adaptarlas al modelo NLG del sistema GeNeSys. Por último, se presentarán las características específicas utilizadas en el entrenamiento de dicho modelo.

3.1. Generación del conjunto de datos de entrenamiento

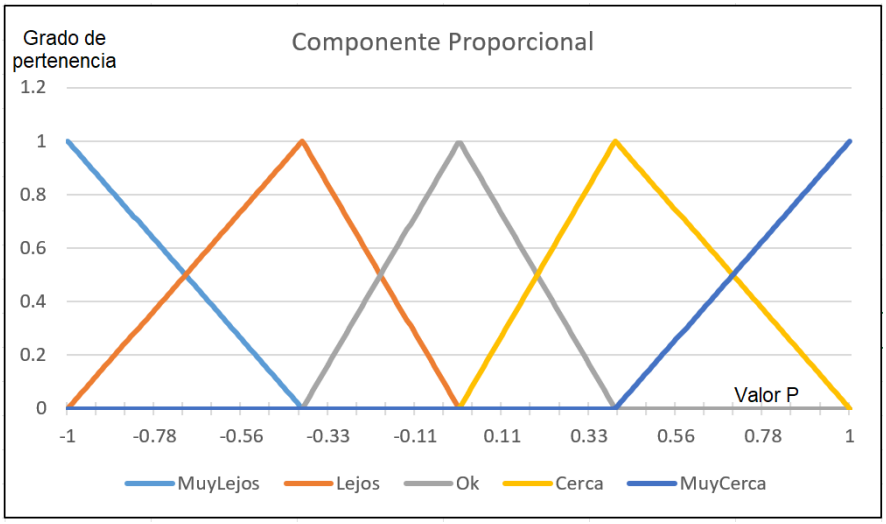
Los robots del sistema GeNeSys que poseen el conocimiento para ejecutar la tarea de seguimiento de pared, actúan como potenciales instructores en la eventual interacción de robots. Estos robots generan las variables descritas en la sección 2.3. Cada rango de estos valores determina el universo de discurso, donde interactúan los conjuntos difusos y se ejecutan los pasos del controlador Mamdani.

Los conjuntos difusos son triangulares para todos sus componentes, por lo tanto, sus grados de pertenencia se calculan mediante la ecuación 2.1.

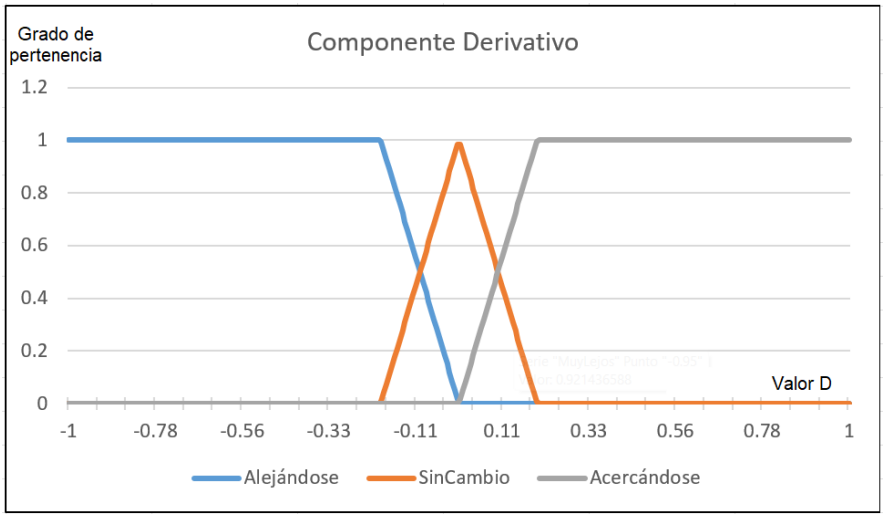
3.1.1. Definición de conjuntos difusos

Cada uno de los términos lingüísticos asociados a los valores P, D, L y R guarda una relación semántica con el propósito de cada uno de ellos en el proceso de generalización y clusterización del GeNeSyS (descritos en la sección 2.3). En la figura 3.1a, el valor P gráficamente representado, consta de 5 términos que ilustran de manera ambigua la cercanía o lejanía a la pared. De manera similar, la figura 3.1b muestra el valor D, que indica si el objeto se está acercando o alejando.

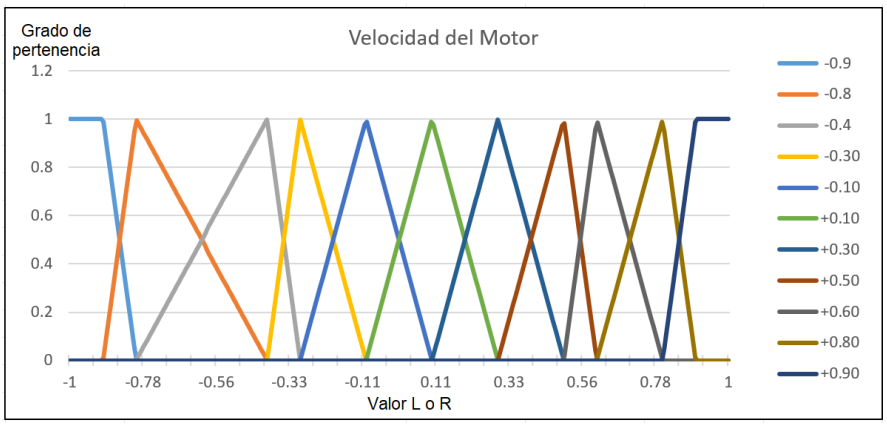
En cuanto a los valores L y R, se definen 11 conjuntos difusos, cuyos términos lingüísticos se representan de forma práctica en la figura 3.1c, abarcando valores discretos que van desde -0.9 hasta +0.9. Estos términos lingüísticos están relacionados con la velocidad angular de los motores y su ambigüedad semántica se puede apreciar en la tabla 3.1. Cada término utilizado en los nombres de los conjuntos difusos puede incluir expresiones y modismos particulares que se aprenden a través de la experiencia y la interacción de los individuos.



(A) Componente P



(B) Componente D



(C) Salidas L y R

FIGURA 3.1. Representación gráfica de conjuntos difusos de todos los componentes.

TABLA 3.1. Valores discretos y términos lingüísticos correspondientes a la figura 3.1c.

Valor	Término lingüístico
-0.9	Muy rápido hacia atrás
-0.8	Bastante rápido hacia atrás
-0.4	Más o menos rápido hacia atrás
-0.3	Despacito hacia atrás
-0.1	Muy lento hacia atrás
0.1	Muy lento hacia delante
0.3	Despacito hacia delante
0.5	Medio rápido hacia delante
0.6	Rápidamente hacia delante
0.8	Bastante rápido hacia delante
0.9	Muy rápido hacia delante

3.1.2. Memoria asociativa difusa

De acuerdo a la cantidad de conjuntos difusos en los componentes P y D, existen 15 posibles combinaciones que activan una regla específica mediante la ecuación 2.2, siendo estas reglas parte de un conocimiento experto que de forma nativa requiere el FIS Mamdani [23]. Los valores de pertenencia de P y D generan la activación de una regla posible como se puede ver en la tabla 3.2. Por ejemplo, si un determinado valor P y otro D pertenecen al conjunto "Lejos" y "Acercándose" con grado de pertenencia 1 respectivamente, se activa la regla 0.5 del valor L y 0.1 de R. Al Verificar la tabla 3.1 se observa que el valor L activa la regla "Medio rápido hacia delante" mientras que el valor R activa "Muy lento hacia delante".

TABLA 3.2. Activación de reglas L y R

(A) Reglas componente L

	MuyLejos	Lejos	Ok	Cerca	MuyCerca
Acercándose	0.6	0.5	-0.1	-0.9	-0.3
SinCambio	0.8	0.6	0.3	-0.8	-0.3
Alejándose	0.9	0.8	0.6	0.1	-0.3

(B) Reglas componente R

	MuyLejos	Lejos	Ok	Cerca	MuyCerca
Acercándose	-0.1	0.1	0.6	0.9	-0.3
SinCambio	-0.3	-0.1	0.3	0.8	-0.3
Alejándose	-0.4	-0.3	-0.1	0.5	-0.3

3.1.3. Tabulación de los datos

Implementando Python, se generaron diez mil valores aleatorios para los componentes P y D, distribuidos uniformemente dentro de sus respectivos universos de discurso. Estos valores se utilizaron como las primeras dos características o *features* del conjunto de datos para el entrenamiento. Para computar los valores de

L y R, se creó una instancia de un controlador FIS utilizando la librería skfuzzy. Este controlador recibió todos los valores de P y D como parámetros, los que se denominan "antecedentes" según la librería [24]. Los valores de P y D se asignaron a los conjuntos difusos respectivos de las figuras 3.1a y 3.1b como valores predeterminados. Luego, se calcularon los valores utilizando la ecuación 2.4. El resultado de este cálculo es un valor que forma parte del universo de discurso "consecuente", cuyos conjuntos difusos están representados en la figura 3.1c.

La librería skfuzzy proporciona métodos que permiten obtener información potencialmente útil para el entrenamiento del modelo, como el término lingüístico asociado al respectivo conjunto difuso y su grado de pertenencia. En las tablas 3.4 y 3.5 se muestra la información obtenida de skfuzzy basada en los valores de P, D, L y R de la tabla 3.3.

TABLA 3.3. Valores de P, D, L y R

P	D	L	R
0.0347	0.1795	-0.1579	0.5969
-0.9689	0.8072	0.6261	-0.0919
0.9820	-0.2112	-0.2398	-0.2265
0.0116	-0.0365	0.3073	0.2190
-0.9307	0.2698	0.6200	-0.0840

TABLA 3.4. Información adicional de P y D

Conjunto difuso P	Pertenencia P	Conjunto difuso D	Pertenencia D
ok	0.91	acercandose	0.90
muy lejos	0.95	acercandose	1.0
muy cerca	0.97	alejandose	1.0
ok	0.97	sin cambio	0.82
muy lejos	0.88	acercandose	1.0

TABLA 3.5. Información adicional de L y R

Conjunto difuso L	Pertenencia L	Conjunto difuso R	Pertenencia R
muy lento hacia atrás	0.71	rápidamente hacia delante	0.97
rápidamente hacia delante	0.87	muy lento hacia atrás	0.96
despacito hacia atrás	0.69	despacito hacia atrás	0.63
despacito hacia delante	0.96	despacito hacia delante	0.59
rápidamente hacia delante	0.90	muy lento hacia atrás	0.92

3.2. Preprocesamiento del texto y los datos

De los datos tabulados obtenidos, los componentes P y D se generaron aleatoriamente dentro de rangos específicos con el objetivo de condicionar al FIS y obtener valores con grados de pertenencia superiores a 0.8 para cada conjunto difuso. Esta condición se muestra en la tabla 3.4 y es importante tener en cuenta estas condiciones durante el preprocesamiento de todos los datos numéricos.

3.2.1. Análisis exploratorio de datos

En el análisis de los datos, se observa una clara agrupación de los conjuntos difusos P y D en su distribución. Sin embargo, los valores L y R no presentan la misma tendencia, como se evidencia en la gráfica 3.2. Esta disparidad se debe a la dependencia de los conjuntos L y R con respecto a la inferencia Mamdani, que se explica detalladamente en la sección 2.1. Además, al examinar las agrupaciones en pares de componentes en la misma figura, se puede apreciar la posible existencia de correlaciones entre ellos. Estas correlaciones se visualizan en la figura 3.3 y son coherentes con los comportamientos esperados para una tarea de seguimiento de pared. A partir de estas observaciones, se puede llegar a una conclusión preliminar en la que el FIS Mamdani adopta decisiones más drásticas para alejarse o acercarse a la pared con el motor izquierdo.

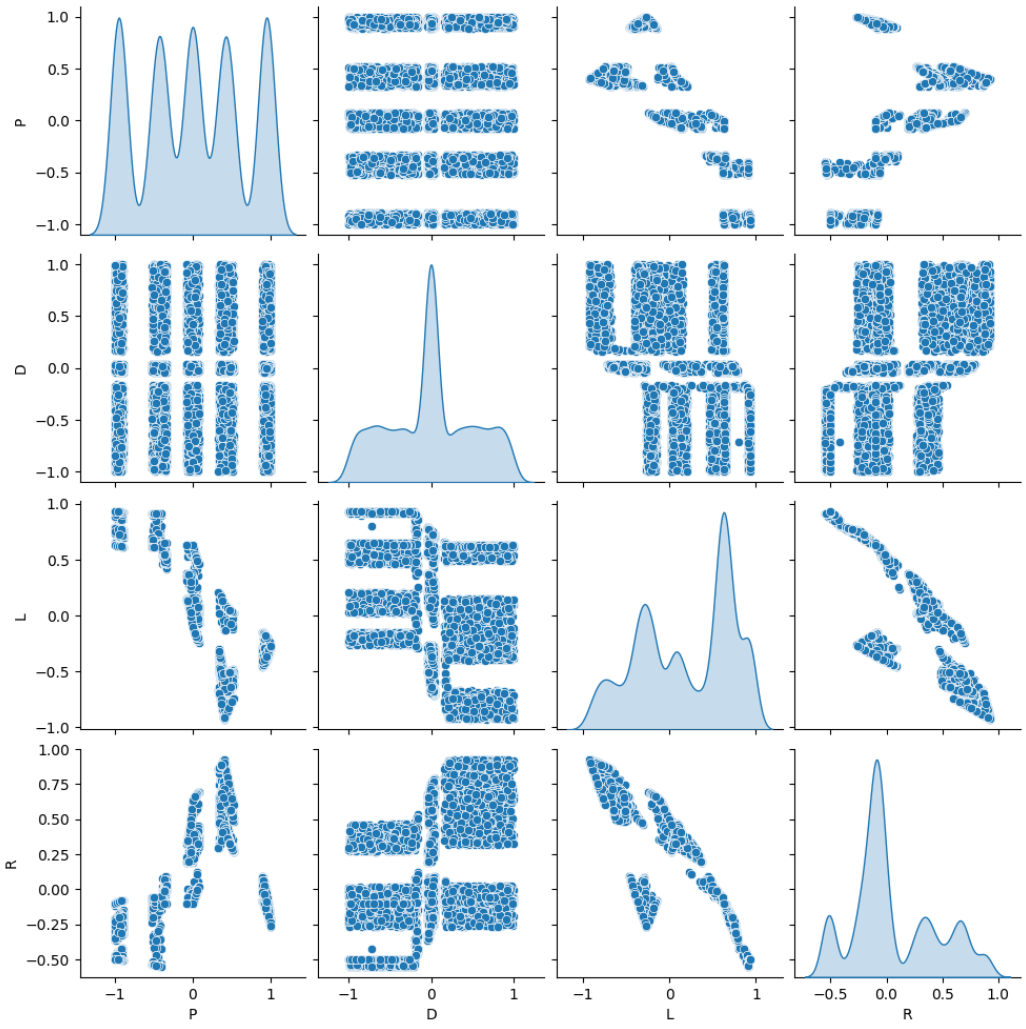


FIGURA 3.2. Distribución de pares de P, D, L y R[1].

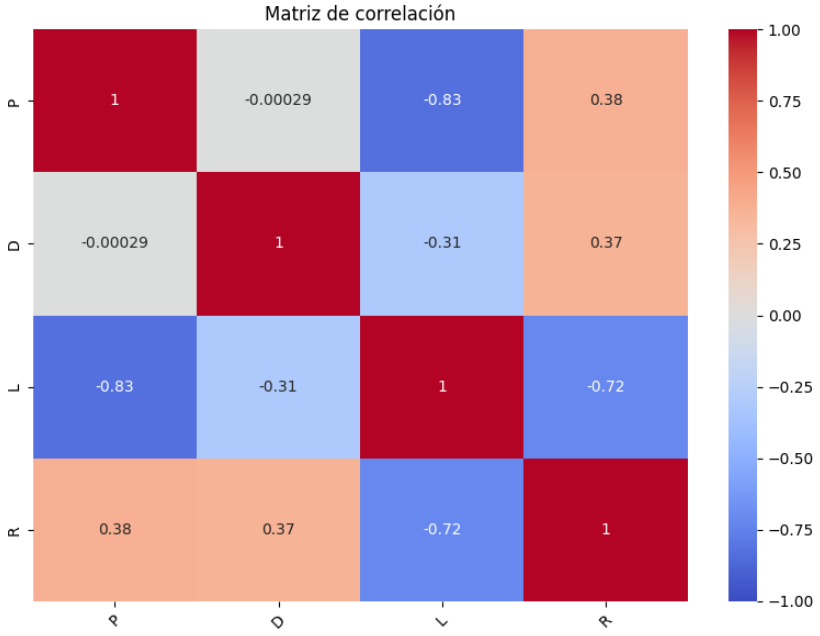


FIGURA 3.3. Correlación de P, D, L y R[2].

3.2.2. Etiquetado de datos y asignación de frases objetivo

Es factible llevar a cabo la categorización de cada uno de los registros presentes en el conjunto de datos mediante la aplicación de las reglas activadas en la memoria asociativa difusa, detalladas exhaustivamente en las tablas 3.2. En aras de establecer un proceso automático de asignación de frases, se han asignado categorías específicas que se encuentran expuestas en la tabla 3.6 y actúan como etiquetas descriptivas. En la figura 3.4 se puede visualizar la distribución de los datos por las categorías previamente expuestas.

Siguiendo el ejemplo expuesto en la sección 3.1.2, donde se considera P como "Lejos", D como "Acercándose", L como "Medio rápido hacia delante" y R "Muy lento hacia delante", es posible concatenar estos cuatro conjuntos difusos para obtener una estructura de frase completa. Esta estructura se verá complementada con los demás componentes de una oración estructurada, resultando en la siguiente formulación: "Cuándo estés lejos de la pared y te estes acercando a ella, gira tu rueda izquierda medio rápido hacia delante y tu rueda derecha medio lento hacia delante". Mediante este enfoque, se logra generar una frase que puede ser considerada como una potencial instrucción verbalizada, susceptible de ser emitida por el robot instructor perteneciente al sistema GeNeSys. Asimismo, es importante destacar que las frases establecidas como etiquetas en las quince categorías correspondientes, pueden ser consultadas en la tabla 3.7 y serán utilizadas en calidad de corpus textual para la herramienta de tokenización integrada en el framework Keras con relleno de secuencias. De este modo, se viabiliza un enfoque integral que favorece la tarea de procesamiento y análisis del lenguaje natural en el contexto planteado.

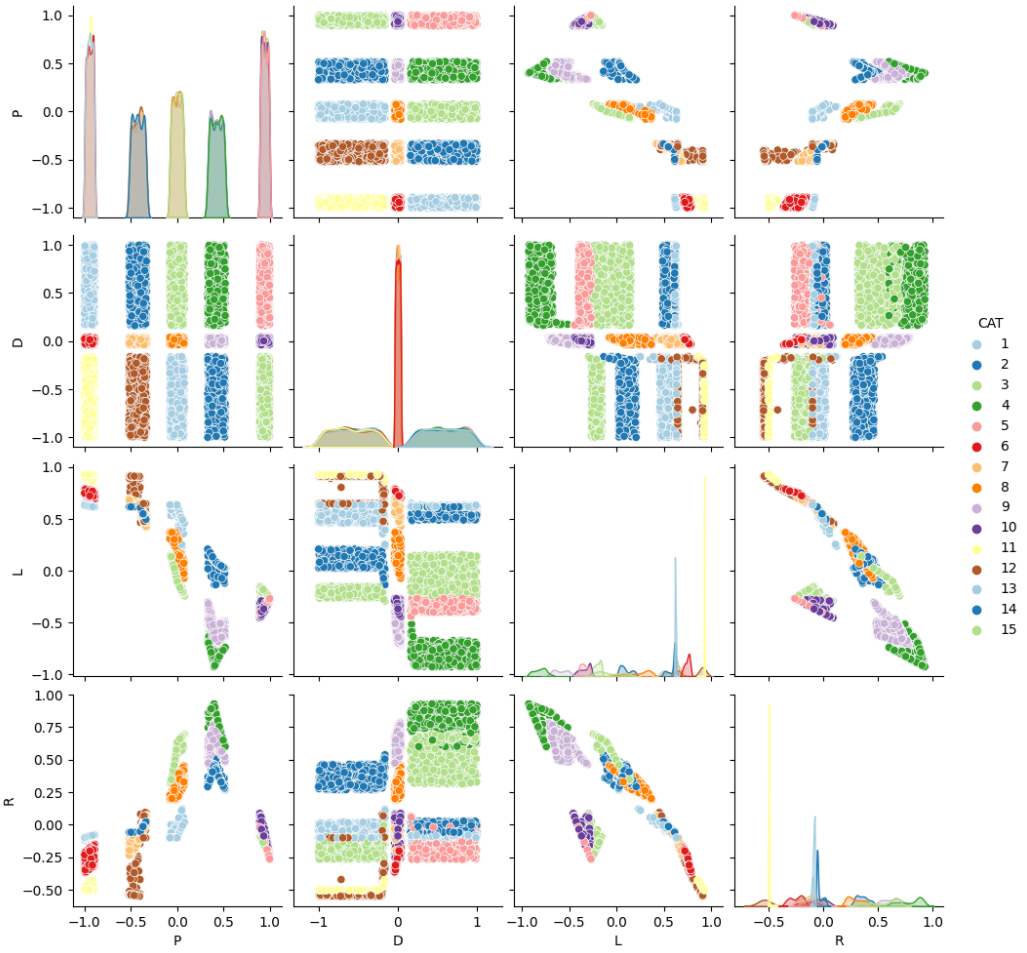


FIGURA 3.4. Distribución de datos por categorías.[3].

TABLA 3.6. Datos de los conjuntos P, D y categoría

Conjunto P	Conjunto D	Categoría
Muy lejos	Acercándose	1
Lejos	Acercándose	2
Ok	Acercándose	3
Cerca	Acercándose	4
Muy cerca	Acercándose	5
Muy lejos	Sin cambio	6
Lejos	Sin cambio	7
Ok	Sin cambio	8
Cerca	Sin cambio	9
Muy cerca	Sin cambio	10
Muy lejos	Alejándose	11
lejos	Alejándose	12
ok	Alejándose	13
Cerca	Alejándose	14
Muy cerca	Alejándose	15

TABLA 3.7. Frases por categoría

Categoría	Frase
1	Cuando estés muy lejos de la pared, pero te estás acercando a ella, gira tu rueda izquierda rápidamente hacia delante, y gira tu rueda derecha muy lento hacia atrás.
2	Cuando estés lejos de la pared, pero te estás acercando a ella, gira tu rueda izquierda medio rápido hacia delante, y gira tu rueda derecha muy lento hacia delante.
3	Cuando estés a la distancia requerida, pero te estás acercando a la pared, gira tu rueda izquierda muy lento hacia atrás, y gira tu rueda derecha rápidamente hacia delante.
4	Cuando estés cerca a la pared, y te estás acercando más a ella, gira tu rueda izquierda muy rápido hacia atrás, y gira tu rueda derecha muy rápido hacia delante.
5	Cuando estés muy cerca a la pared, y te estás acercando aun más a ella, gira tu rueda izquierda despacito hacia atrás, y gira tu rueda derecha despacito hacia atrás.
6	Cuando estés muy lejos de la pared, pero ni te alejas más ni te acercas a ella, gira tu rueda izquierda bastante rápido hacia delante, y gira tu rueda derecha despacito hacia atrás.
7	Cuando estés lejos de la pared, pero ni te alejas más ni te acercas a ella, gira tu rueda izquierda rápidamente hacia delante, y gira tu rueda derecha muy lento hacia atrás.
8	Cuando estés a la distancia requerida, y ni te alejas ni te acercas a ella, gira tu rueda izquierda despacito hacia delante, y gira tu rueda derecha despacito hacia delante.
9	Cuando estés cerca a la pared, pero ni te acercas más ni te alejas de ella, gira tu rueda izquierda bastante rápido hacia atrás, y gira tu rueda derecha bastante rápido hacia delante.
10	Cuando estés muy cerca a la pared, pero ni te acercas más ni te alejas de ella, gira tu rueda izquierda despacito hacia atrás, y gira tu rueda derecha despacito hacia atrás.
11	Cuando estés muy lejos de la pared, y te estás alejando aún más de ella, gira tu rueda izquierda muy rápido hacia delante y gira tu rueda derecha más o menos rápido hacia atrás.
12	Cuando estés lejos de la pared, y te estás alejando más de ella, gira tu rueda izquierda bastante rápido hacia delante, y gira tu rueda derecha despacito hacia atrás.
13	Cuando estés a la distancia requerida, pero te estás alejando de la pared, gira tu rueda izquierda rápidamente hacia delante, y gira tu rueda derecha muy lento hacia atrás.
14	Cuando estés cerca a la pared, pero te estás alejando de ella, gira tu rueda izquierda muy lento hacia delante, y gira tu rueda derecha medio rápido hacia delante.
15	Cuando estés muy cerca a la pared, pero te estás alejando de ella, gira tu rueda izquierda despacito hacia atrás, y gira tu rueda derecha despacito hacia atrás.

3.3. Descripción del modelo implementado

En la biblioteca Skfuzzy, se utiliza la definición de conjuntos difusos antecedentes y consecuentes como un *pipeline* en el procesamiento de los valores P, D, L y R. Este proceso asigna a cada valor de entrada el conjunto difuso al que pertenece, tomando el término con el valor más alto de membresía. Esta característica agrega al modelo de este trabajo una similitud con los modelos co-evolutivos del GeNeSys, específicamente en la inferencia difusa para la generalización de los datos.

Se ha creado un modelo de secuencias básico utilizando *seq2seq* con PyTorch. El codificador consta de una capa de celdas LSTM con 64 neuronas, mientras que el decodificador es idéntico al codificador, con la adición de una capa de *embedding* que recibe las secuencias ya tokenizadas de los términos lingüísticos generados por el FIS. Por último, se incluye una capa densa con un número de neuronas igual al tamaño del vocabulario que entrega las secuencias inferidas. La estructura final del modelo puede visualizarse en la figura 3.5.

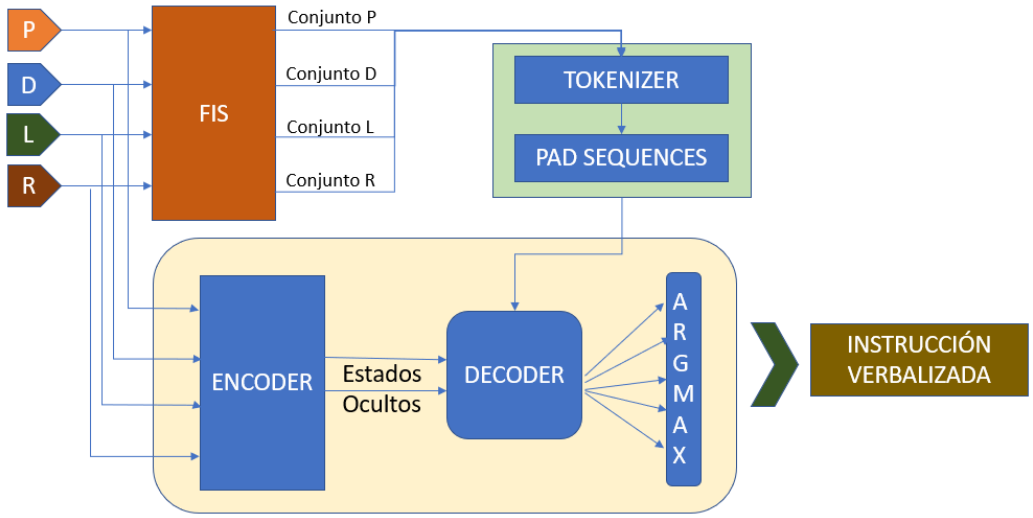


FIGURA 3.5. Arquitectura del modelo NLG del GeNeSys

3.4. Entrenamiento del modelo

Con los datos de entrada y las frases objetivo completas ya preprocesadas en secuencias numéricas, se procede a dividir los datos en conjuntos de entrenamiento (60 %), validación (20 %) y pruebas (20 %) con distribución uniforme de las 15 categorias. Los términos lingüísticos de cada conjunto difuso se consideran características del dataset tabular y se concatenan para generar una secuencia de entrada en el decoder, como se ilustra en la figura 3.5.

El modelo se entrena durante 150 épocas utilizando el optimizador Adam con una tasa de aprendizaje de 0.001 y un tamaño de lote de 64. La eficiencia computacional del modelo se evalúa utilizando un equipo ASUS TUF con procesador Ryzen 3550h sin GPU disponible y la plataforma Google Colaboratory para algunos casos de uso de GPU.

Capítulo 4

Ensayos y resultados

4.1. Software utilizado

Webots es un paquete de software profesional para simulación de robots móviles. Permite crear entornos virtuales en 3D con propiedades físicas, como masa, articulaciones, coeficientes de fricción, entre otros. Los usuarios pueden agregar objetos simples o robots móviles con diferentes esquemas de locomoción (ruedas, patas o vuelo). Además, se pueden equipar con sensores y actuadores, como sensores de distancia, ruedas motrices, cámaras, motores, sensores táctiles, emisores y receptores, entre otros. Los robots pueden programarse individualmente para exhibir el comportamiento deseado.

Webots es especialmente útil para proyectos de investigación y educación relacionados con la robótica móvil. Se ha utilizado en diversas áreas, como prototipado de robots móviles (investigación académica, industria automotriz, aeronáutica, industria de aspiradoras, industria de juguetes, aficionado, entre otros), investigación en locomoción de robots (robots con patas, humanoides, cuadrúpedos, entre otros), investigación en comportamiento adaptativo (algoritmos genéticos, redes neuronales, inteligencia artificial, etc.), enseñanza de robótica (cursos de robótica, programación en C/C++/Java/Python, etc.) y competencias de robots [25].

4.1.1. Implementación del modelo en webots

La implementación (ver Figura 4.1) exhibe al robot inmerso en un entorno donde se llevan a cabo tanto el aprendizaje como la evaluación de su comportamiento en la tarea cóncava. En la sección detalles de la plataforma webots, se define un árbol de nodos que engloba diversos aspectos físicos de los robots a simular (ver Figura 4.2a). Dentro de estos elementos, se encuentra el nodo-Robot tipo supervisor que alberga un controlador Python llamado MasterConcava90-51-wPT. Este ejecuta el método `Recomendaciones()`, que se encuentra en la parte del código del ambiente webots tal como se muestra en la figura 4.2b. El propósito de este método es enviar los valores P, D, L y R al modelo NLG y generar un archivo de extensión .txt con los resultados obtenidos.

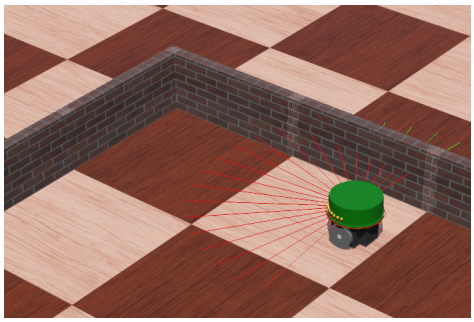
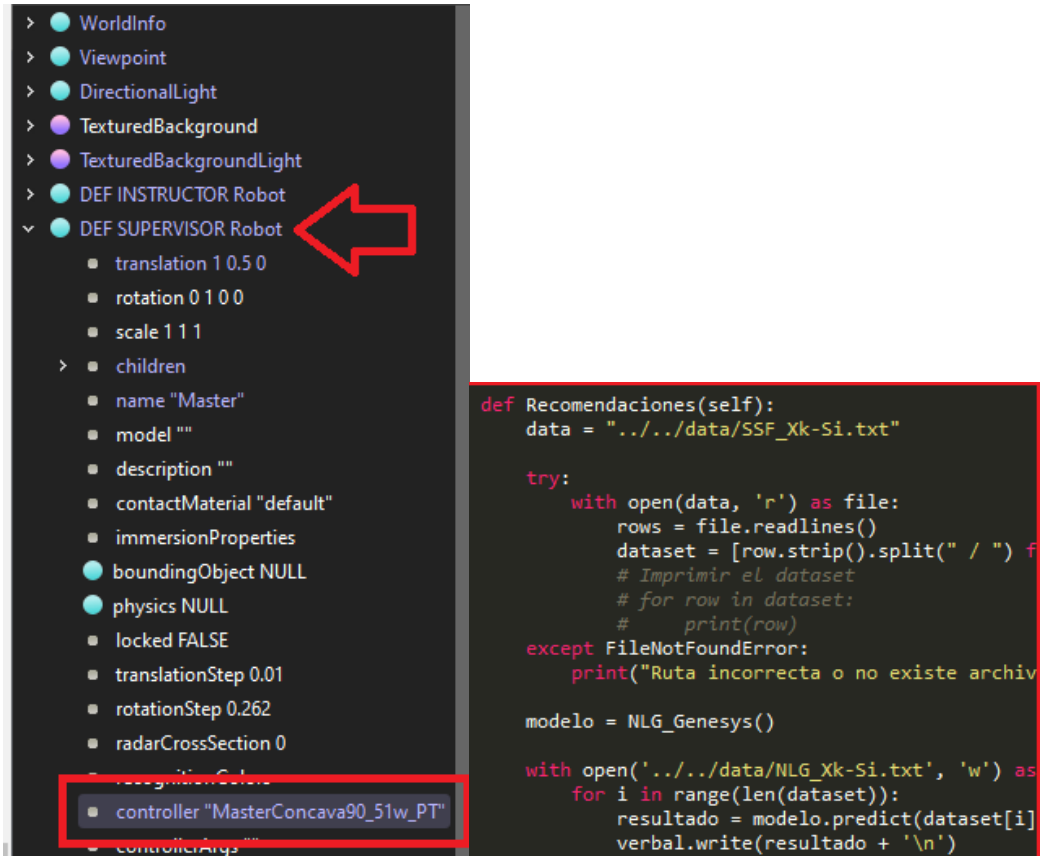


FIGURA 4.1. Escenario de prueba del robot instructor



(A) a) (B) b)

FIGURA 4.2. a.) Nodo supervisor y controlador. b.) Método de activación NLG

4.2. Pruebas unitarias

Los datos utilizados en esta sección fueron extraídos del subconjunto de datos con una distribución de clases uniforme, tal como se ilustra en la figura 4.3 y que fue previamente mencionado en la sección 3.4. Utilizando los mismos criterios, se generan 15 datos arbitrarios que no hicieron parte del mencionado conjunto de datos y corresponden a un registro por categoría (ver tabla 4.1). Durante estas pruebas unitarias, se sometieron a evaluación todas las etapas funcionales del modelo NLG descrito en la sección 3.3.

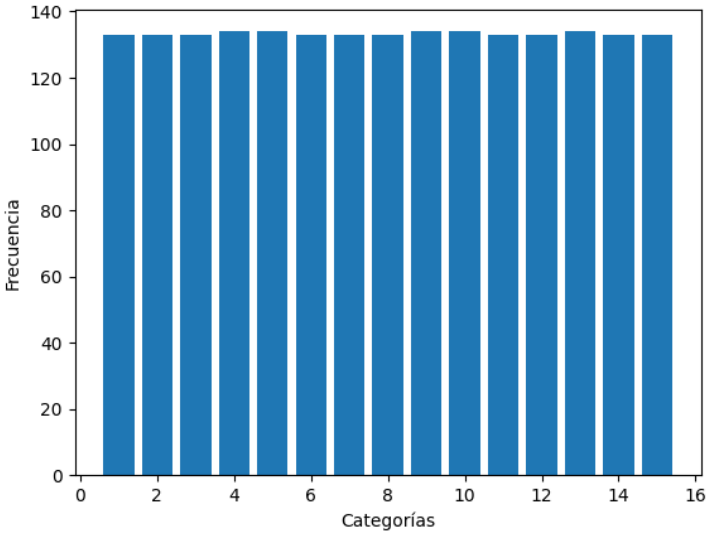


FIGURA 4.3. Distribución de clases en datos de prueba

TABLA 4.1. Valores pruebas unitarias

P	D	L	R	CAT
-0.8970	0.7209	0.5828	-0.0657	1
-0.4765	0.9839	0.5127	0.0745	2
-0.0473	0.9302	-0.0291	0.5409	3
0.4314	0.7532	-0.8686	0.8373	4
0.9873	0.5057	-0.3127	-0.2746	5
-0.8971	-0.0014	0.7669	-0.2669	6
-0.4792	0.0159	0.6159	-0.1090	7
-0.0079	0.0239	0.2634	0.3232	8
0.4102	0.0126	-0.7897	0.7699	9
0.9635	0.0123	-0.3596	-0.1754	10
-0.9777	-0.9732	0.8963	-0.3963	11
-0.5071	-0.6878	0.8179	-0.3179	12
0.0798	-0.9896	0.5002	0.0197	13
0.4882	-0.3743	0.0412	0.3824	14
0.9472	-0.8104	-0.2648	-0.2296	15

4.2.1. Resultados por etapas del NLG

Cada uno de los valores correspondientes a las diferentes categorías fue sometido al proceso del FIS con el objetivo de obtener los conjuntos difusos que presentaron el mayor grado de pertenencia. Estos conjuntos difusos se entregan en forma de una lista que se concatena para formar una única frase que no necesariamente constituye una estructura coherente. Posteriormente, esta frase se transforma en una secuencia numérica que se utilizará como entrada para el decodificador del modelo de *seq2seq*. Los conjuntos difusos de cada categoría y las respectivas secuencias se observan en la tabla 4.2.

TABLA 4.2. Datos entregados por el FIS - Secuencias Tokenizer

Conjuntos difusos FIS	**Secuencias Tokenizer**
muy lejos, acercándose, rápidamente hacia delante, muy lento hacia atrás	14, 27, 32, 4, 15, 14, 26, 4, 9
lejos, acercándose, medio rápido hacia delante, muy lento hacia delante	27, 36, 23, 4, 15, 14, 26, 4, 15
ok, acercándose, muy lento hacia atrás, medio rápido hacia delante	14, 26, 4, 9, 36, 23, 4, 15
cerca, acercándose, muy rápido hacia atrás, bastante rápido hacia delante	25, 14, 23, 4, 9, 33, 23, 4, 15
muy cerca, acercándose, despacito hacia atrás, despacito hacia atrás	14, 25, 19, 4, 9, 19, 4, 9
muy lejos, sin cambio, bastante rápido hacia delante, despacito hacia atrás	14, 27, 33, 23, 4, 15, 19, 4, 9
lejos, sin cambio, rápidamente hacia delante, muy lento hacia atrás	27, 32, 4, 15, 14, 26, 4, 9
ok, sin cambio, despacito hacia delante, despacito hacia delante	19, 4, 15, 19, 4, 15
cerca, sin cambio, bastante rápido hacia atrás, bastante rápido hacia delante	25, 33, 23, 4, 9, 33, 23, 4, 15
muy cerca, sin cambio, más o menos rápido hacia atrás, muy lento hacia atrás	14, 25, 24, 39, 40, 23, 4, 9, 14, 26, 4, 9
muy lejos, alejándose, muy rápido hacia delante, más o menos rápido hacia atrás	14, 27, 14, 23, 4, 15, 24, 39, 40, 23, 4, 9
lejos, alejándose, bastante rápido hacia delante, despacito hacia atrás	27, 33, 23, 4, 15, 19, 4, 9
ok, alejándose, medio rápido hacia delante, muy lento hacia delante	36, 23, 4, 15, 14, 26, 4, 15
cerca, alejándose, muy lento hacia delante, despacito hacia delante	25, 14, 26, 4, 15, 19, 4, 15
muy cerca, alejándose, despacito hacia atrás, despacito hacia atrás	14, 25, 19, 4, 9, 19, 4, 9

En última instancia, se lleva a cabo un análisis de las frases generadas para cada categoría. Es importante destacar que las frases generadas, si bien carecen de los signos de puntuación y otras convenciones gramaticales propias del idioma español, arrojan resultados idénticos a los presentados en la tabla 3.7. A continuación se presentan las frases correspondientes a cada categoría generadas por el modelo NLG:

1. Cuando estés muy lejos de la pared pero te estás acercando a ella gira tu rueda izquierda rápidamente hacia delante y gira tu rueda derecha muy lento hacia atrás.
2. Cuando estés lejos de la pared pero te estás acercando a ella gira tu rueda izquierda medio rápido hacia delante y gira tu rueda derecha muy lento hacia delante.
3. Cuando estés a la distancia requerida pero te estás acercando a la pared gira tu rueda izquierda muy lento hacia atrás y gira tu rueda derecha rápidamente hacia delante.

4. Cuando estés cerca a la pared y te estás acercando más a ella gira tu rueda izquierda muy rápido hacia atrás y gira tu rueda derecha muy rápido hacia delante.
5. Cuando estés muy cerca a la pared y te estás acercando aún más a ella gira tu rueda izquierda despacito hacia atrás y gira tu rueda derecha despacito hacia atrás.
6. Cuando estés muy lejos de la pared pero ni te alejas más ni te acercas a ella gira tu rueda izquierda bastante rápido hacia delante y gira tu rueda derecha despacito hacia atrás.
7. Cuando estés lejos de la pared pero ni te alejas más ni te acercas a ella gira tu rueda izquierda rápidamente hacia delante y gira tu rueda derecha muy lento hacia atrás.
8. Cuando estés a la distancia requerida y ni te alejas ni te acercas a ella gira tu rueda izquierda despacito hacia delante y gira tu rueda derecha despacito hacia delante.
9. Cuando estés cerca a la pared pero ni te acercas más ni te alejas de ella gira tu rueda izquierda bastante rápido hacia atrás y gira tu rueda derecha bastante rápido hacia delante.
10. Cuando estés muy cerca a la pared pero ni te acercas más ni te alejas de ella gira tu rueda izquierda despacito hacia atrás y gira tu rueda derecha despacito hacia atrás.
11. Cuando estés muy lejos de la pared y te estás alejando aun más de ella gira tu rueda izquierda muy rápido hacia delante y gira tu rueda derecha más o menos rápido hacia atrás.
12. Cuando estés lejos de la pared y hacia estás alejando más y ella gira tu rueda izquierda bastante lejos hacia delante y gira tu rueda derecha despacito hacia atrás.
13. Cuando estés a la distancia requerida pero te estás alejando de la pared gira tu rueda izquierda rápidamente hacia delante y gira tu rueda derecha muy lento hacia atrás.
14. Cuando estés cerca a la pared pero te estás alejando de ella gira tu rueda izquierda muy lento hacia delante y gira tu rueda derecha medio rápido hacia delante.
15. Cuando estés muy cerca a la pared pero te estás alejando de ella gira tu rueda izquierda despacito hacia atrás y gira tu rueda derecha despacito hacia atrás.

4.2.2. Errores en las secuencias generadas en el dataset de test

El conjunto de pruebas utilizado se compone de 2000 ejemplos que fueron sometidos al modelo para la generación de secuencias. A través del análisis de las frases generadas, se detectaron errores que afectaron tanto a palabras individuales como a la estructura general de las secuencias. En particular, se identificó un

total de 11 secuencias con errores, distribuidas de la siguiente manera: 7 secuencias pertenecen a la categoría 12, 3 secuencias corresponden a la categoría 9 y 1 secuencia a la categoría 8.

De las 7 secuencias pertenecientes a la categoría 12 que presentaron errores, se observó que 5 de ellas mostraron una mayor probabilidad de encontrar el token "lejos" en lugar del correspondiente "lento". Ambas palabras comparten la característica de representar una condición o estado opuesto a la rapidez, ya sea en términos de distancia o movimiento. Sin embargo, esta variación en la elección del token genera confusión al definir la instrucción relacionada con el movimiento de la rueda correspondiente. En las dos secuencias restantes, se agregaron los tokens "pero" y "lento", que resultaron en frases confusas o redundantes. En la tabla 4.3 se pueden apreciar las palabras incorrectas o añadidas, resaltadas en color rojo. Por otro lado, en las frases pertenecientes a la categoría 9, se observa una irregularidad en la instrucción para girar la rueda derecha. Finalmente, la única frase de la categoría 8 no especifica de manera concreta el componente de acercamiento a la pared.

TABLA 4.3. Secuencias erróneas

CAT	Secuencias
12	cuando estés lejos de la pared y te estás alejando más de ella gira tu rueda izquierda bastante lejos hacia delante y gira tu rueda derecha despacito hacia atrás
12	cuando estés lejos de la pared pero te estás alejando más de ella gira tu rueda izquierda tu rápido hacia delante y gira tu rueda derecha despacito hacia atrás lento
8	cuando estés a la distancia requerida y ni te acercas ni te rápidamente a ella gira tu rueda izquierda despacito hacia delante y gira tu rueda derecha despacito hacia delante
9	cuando estés cerca a la pared pero ni te acercas más ni te alejas de ella gira tu rueda izquierda bastante rápido hacia atrás y gira tu hacia derecha bastante rápido hacia delante

4.3. Certificación en individuo

Se selecciona aleatoriamente un individuo de la sociedad de robots que ya tenga el conocimiento necesario para llevar a cabo la tarea de seguimiento de pared, independientemente de los parámetros de sus sistemas conexionistas. Según [5], la comunicación entre los individuos se realiza mediante archivos planos que contienen instrucciones numéricas P, D, L, R; estas instrucciones son depositadas por el robot instructor en un directorio específico, donde es leído por el robot aprendiz para iniciar su proceso de aprendizaje. La Figura 4.4 muestra un fragmento del archivo plano generado por el robot instructor que utiliza para certificar el modelo NLG.

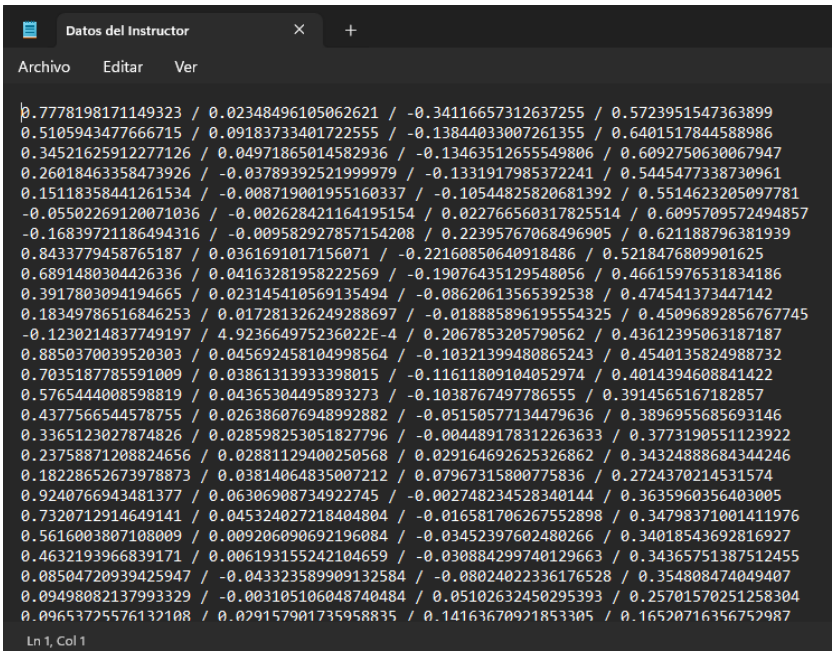


FIGURA 4.4. Archivo plano de robot instructor

4.3.1. Distribución de los datos del instructor

El instructor genera 47 registros cuyos valores, a diferencia de los datos generados en este trabajo, no tienen estrictamente grados de pertenencia mayores a 0,8. Por lo tanto, como se observa en la figura 4.5, su distribución es significativamente diferente, pues contiene datos que no pertenecen a ninguna de las categorías visualizadas en la figura 3.4 y solo una minoría de ellos están clasificados en las categorías 8 y 9.

4.3.2. Resultados con los datos del instructor

Es importante destacar que una considerable cantidad de datos generados por el robot instructor no presenta las mismas características que los datos utilizados para entrenar el modelo NLG. Como resultado, muchas de las secuencias generadas carecen de coherencia y se alejan significativamente de la estructura canónica observada con respecto a las frases de la tabla 3.7. Sin embargo, a pesar de esta variabilidad de los valores de entrada, se pueden identificar subconjuntos de secuencias que mantienen una coherencia interna y siguen la estructura deseada.

En el análisis de los registros generados por el robot instructor, se identificaron un total de 32 secuencias diferentes. De los 48 registros totales, se encontró que 9 de ellos cumplieron con los criterios establecidos y pertenecieron a la categoría 14, mientras que otras 2 secuencias se clasificaron en la categoría 8. Sin embargo, la mayoría de las frases restantes carecían de coherencia y presentaron errores en varios niveles, como la repetición de palabras, la inclusión de términos incorrectos o una estructura gramatical completamente incomprensible. Estos resultados destacan la variabilidad en la calidad y precisión de las secuencias generadas por el modelo NLG en relación con las expectativas establecidas.

A continuación se enumeran algunas de las secuencias erróneas generadas desde los datos del instructor. Allí se evidencian las falencias del uso del FIS Mamdani con respecto al ANFIS del robot del GeNeSys:

1. cuando estés muy a a la pared te estás te acercando rápido estás gira ella de izquierda gira tu rueda y despacito hacia atrás y gira tu rueda muy de hacia
2. cuando estés a a requerida medio y ni te acercando a te distancia a ella gira tu muy izquierda derecha la estás rueda gira tu derecha rápidamente despacito rueda
3. cuando cuando derecha más a ella delante despacito estás pero lejos ni estás a acercando rueda tu gira hacia de hacia de tu la izquierda gira hacia hacia derecha de
4. atrás cuando acercando lejos la a y te estás pero lejos rápido estás tu tu rueda izquierda tu rueda hacia estás y gira tu gira tu rueda hacia ni
5. atrás cuando lejos rápido la a pared te estás pero rápidamente rápido estás tu rueda ella estés alejando hacia te aún y tu rueda derecha de acercando hacia estás

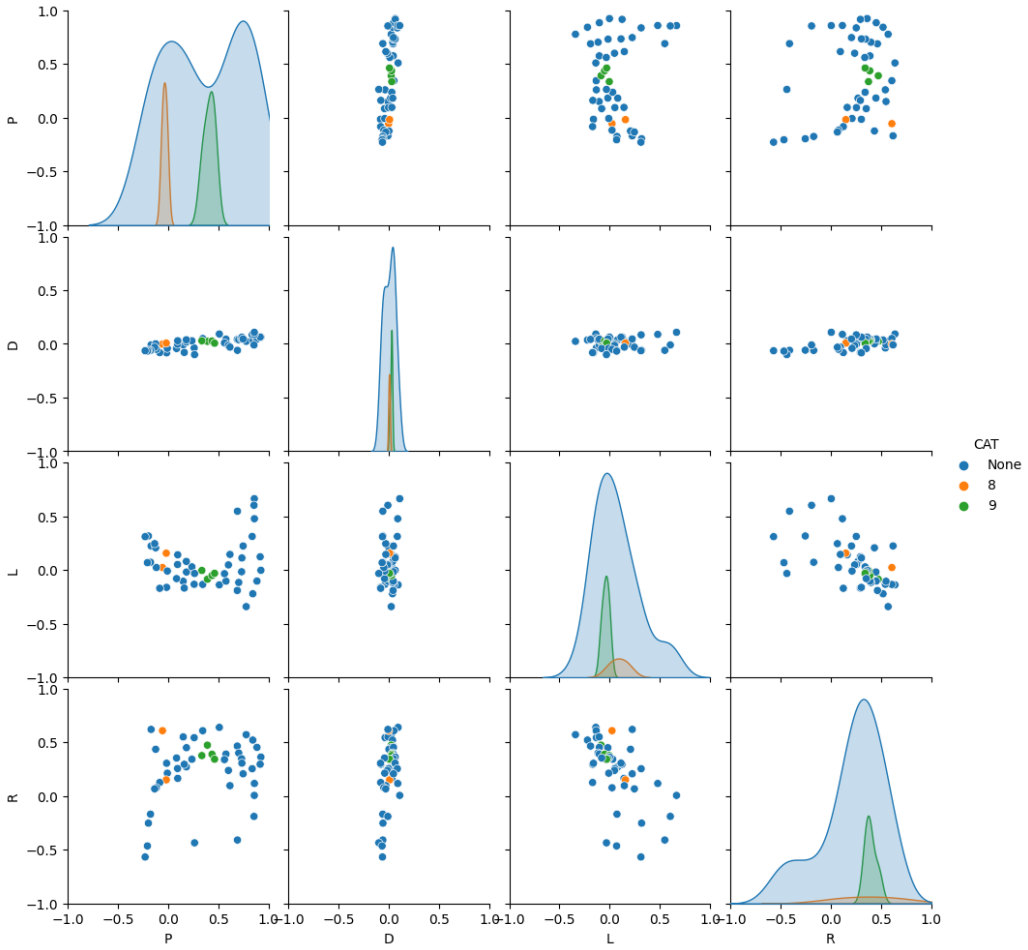


FIGURA 4.5. Distribución de datos del robot instructor

4.4. Métricas

En el marco de este trabajo, se utilizan métricas fundamentales para evaluar el desempeño del modelo de generación de lenguaje natural. La exactitud (*accuracy*) y la pérdida (*loss*) proporcionan una visión cuantitativa de la calidad de las predicciones y la capacidad de ajuste del modelo a los datos. Dado que las frases siguen una estructura canónica en todas sus categorías y la cantidad de datos está balanceada, se evaluarán las clasificaciones del vocabulario utilizando las métricas mencionadas anteriormente.

4.4.1. Exactitud y pérdida en entrenamiento, validación y pruebas

La métrica de exactitud, o *accuracy*, es una medida comúnmente utilizada en problemas de clasificación para determinar la proporción de predicciones correctas respecto al total de predicciones realizadas [26]. En el caso del modelo NLG desarrollado, esta métrica se aplica para evaluar la precisión en la generación de secuencias numéricas, comparando las secuencias generadas con las secuencias de referencia.

Por otro lado, la métrica de pérdida, o *loss*, es una medida que indica la diferencia entre el valor predicho por el modelo y el valor real. El objetivo es minimizar esta pérdida, ya que un valor menor indica un mejor ajuste del modelo a los datos de entrenamiento [27].

De acuerdo al número de épocas de entrenamiento mencionado en la sección 3.4, se visualiza el comportamiento de *accuracy* y *loss* en la figura 4.6 donde se incluyen también resultados en el conjunto de datos de validación. Finalmente, en la tabla 4.4 se visualizan las métricas obtenidas a partir del conjunto de datos de entrenamiento, validación y pruebas generados por el FIS.

TABLA 4.4. Métricas de entrenamiento y pruebas

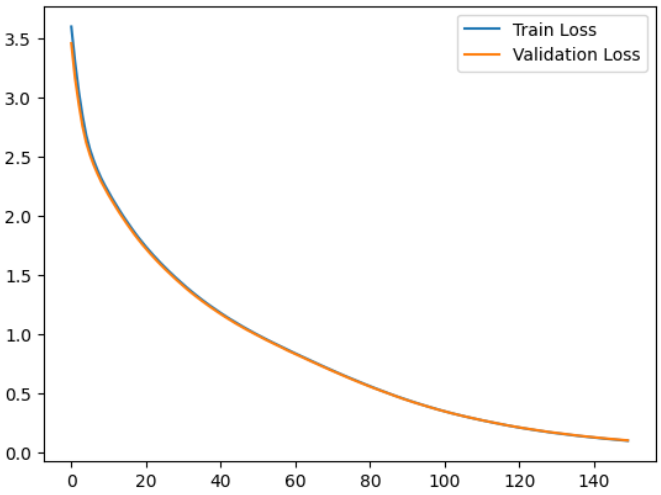
ETAPA	Loss	Accuracy (%)
Entrenamiento/Validación	0.1039/0.1067	99,77
Pruebas	0.1027	99,87

4.5. Comparación con investigaciones relacionadas

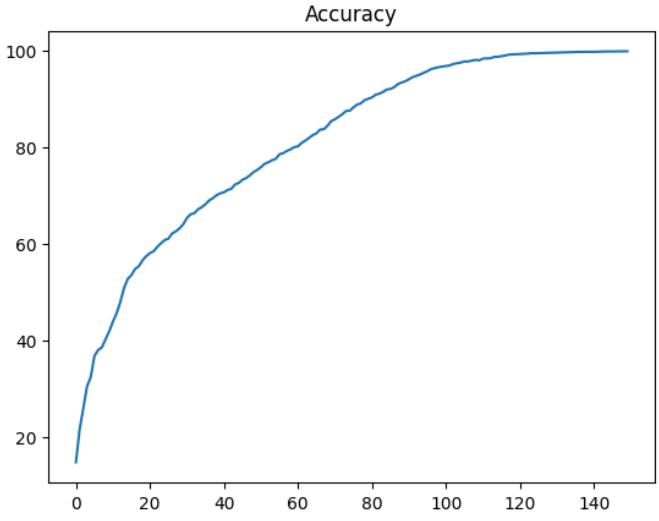
Al comparar diferentes enfoques, es importante tener en cuenta varios aspectos que se derivan de los objetivos planteados en este trabajo. El modelo de generación de lenguaje natural desarrollado en este estudio presenta características distintivas en comparación con los trabajos mencionados en la sección 1.4. En primer lugar, no requiere un preprocesamiento extenso de los datos. A diferencia de los enfoques anteriores, el modelo NLG utilizado en este estudio permite utilizar los datos obtenidos del FIS directamente, sin necesidad de preprocesamiento, tanto para el ingreso al modelo *seq2seq* como durante su implementación. Además, los datos obtenidos del ANFIS se entregaron al modelo en el mismo rango que los datos del FIS. Esto contrasta con los modelos del estado del arte, donde se requiere la creación de estructuras de entidades, grafos lógicos o modelos pre-operativos antes de generar las secuencias correspondientes. Sin embargo, es

importante destacar que las secuencias generadas en este trabajo siguen un patrón gramatical más limitado en comparación con los resúmenes y descripciones generados por otros enfoques.

En general, los demás trabajos en este campo incorporan conocimiento experto tanto en la extracción de los datos de entrada como en su preprocesamiento. Por ejemplo, el modelo mencionado en la sección 1.4.1 utiliza un esquema de grafo lógico que implica la toma de decisiones diseñadas por expertos en sistemas de fraude. Este enfoque guarda similitud con el FIS, donde el diseño de conjuntos difusos puede considerarse como conocimiento experto que contribuye a maximizar la probabilidad de las secuencias en el decodificador del modelo de generación de lenguaje natural.



(a)



(b)

FIGURA 4.6. a) *Loss* en entrenamiento y validación. b) *Accuracy* en entrenamiento.

En cuanto a la complejidad del modelo neuronal empleado, en NLG para el GeNeSys es bastante inferior, no utiliza capas de atención o *transformers*. Por ende su eficiencia computacional en entrenamiento e inferencia ayuda a que el proceso de transmisión de conocimiento no sea fuertemente impactado en la interacción de los individuos GeNeSys.

4.6. Impacto del ANFIS en el modelo NLG

El ANFIS, como el sistema generalizador del GeNeSys, recibe instrucciones de otro robot para iniciar su interacción con el entorno, donde se entrenan sus parámetros antecedentes y consecuentes [5]. En la figura 4.7, se pueden visualizar los conjuntos difusos P y D que corresponden al robot instructor.

Por otro lado, el robot aprendiz, en su proceso de aprendizaje, interactúa con el entorno tratando de asimilar las recomendaciones del robot instructor. Dicho aprendizaje es independiente de la topología de los ANFIS, pues pueden ser diferentes en cada robot, y por esto la cantidad y distribución de los conjuntos difusos en el robot aprendiz es muy diferente, como se muestra en la figura 4.8.

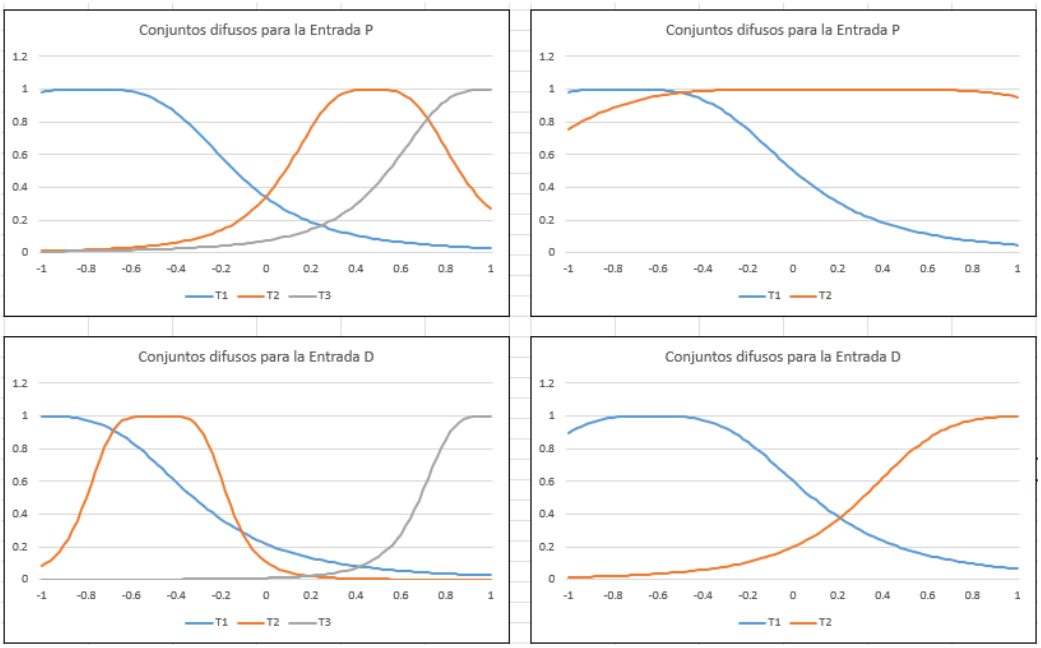


FIGURA 4.7. Conocimiento ANFIS instructor

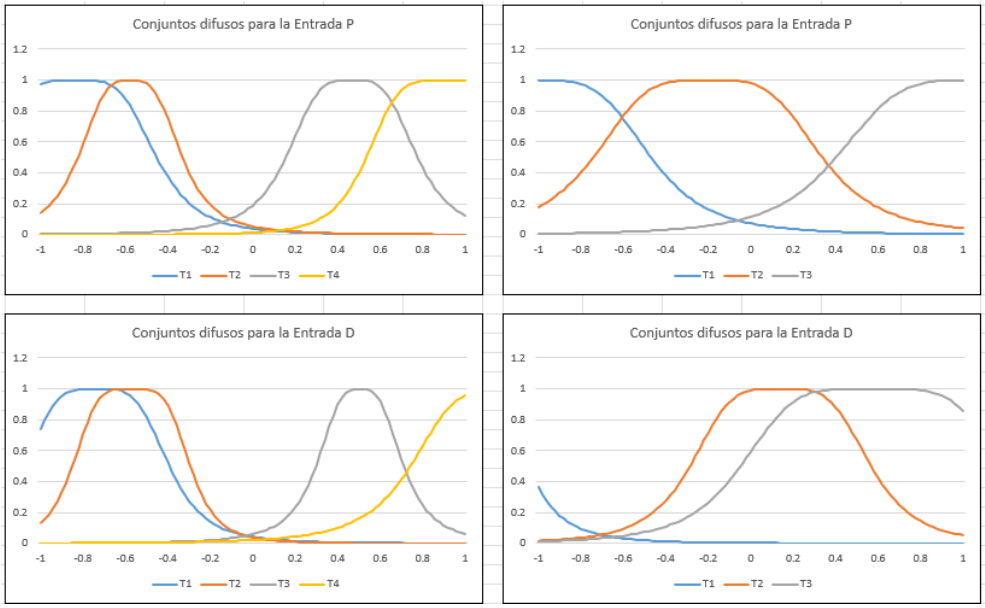


FIGURA 4.8. Conocimiento ANFIS aprendiz

El cambio evidente en los conjuntos difusos de cada individuo confirma la propiedad adaptativa del ANFIS, que provoca cambios significativos que trascienden el contexto del FIS implementado. En consecuencia, la generación de frases correctas en el modelo NLG dependerá de si los datos se ajustan a las distribuciones implementadas en el FIS Mamdani. De lo contrario, como ocurrió en la gran mayoría de los registros, las secuencias serán incomprensibles.

Capítulo 5

Conclusiones

A continuación se presentan las conclusiones obtenidas en la implementación del NLG de la investigación realizada además de listar ciertos trabajos futuros que podrán nacer producto de los resultados.

5.1. Conclusiones generales

Este trabajo ha permitido la combinación de lógica difusa con un sistema de aprendizaje profundo para generar secuencias que representan potenciales instrucciones verbales para un robot que desea enseñarle su comportamiento a otro. Si bien la mayoría de las secuencias generadas con el FIS fueron correctas, la propiedad adaptativa del ANFIS introdujo diferencias significativas en la distribución del conocimiento para la tarea de seguimiento de pared, derivando en la generación de secuencias incomprensibles. Por lo tanto, es importante considerar que el modelo de aprendizaje profundo utilizado debería tener propiedades adaptativas que le permitan ajustarse a la diversidad de comportamientos de los sistemas conexionistas presentes en el GeNeSys.

Al parecer, la incongruencia en las frases generadas con datos del ANFIS podría ser causada parcialmente por la falta de ejemplos en muchas combinaciones de P, D, L y R que cumplen el criterio de grados de pertenencia mayor a 0.8. Este principio se consideró necesario ya que de lo contrario las respuestas implicarían la combinación de reglas difusas del FIS Mamdani, es decir, acarrearía el comportamiento del controlador original. Por lo tanto, el comportamiento del ANFIS se vería totalmente suprimido en la interacción con el modelo para la generación de expresiones lingüísticas. Sin embargo, esta falta de ejemplos en ciertas combinaciones de P, D, L y R afecta la capacidad del modelo para generar frases coherentes.

Relacionar números decimales con texto ha sido un desafío considerable. Las combinaciones de texto y números generalmente se han realizado en el contexto de entrada entera y salida vectorial utilizando capas de *embedding*. Sin embargo, mediante el uso del FIS y los términos lingüísticos, se logró establecer una relación aproximada que ayudó en la generación de estas secuencias. Es importante destacar que en el contexto del ANFIS, estos términos lingüísticos son impredecibles.

Por otra parte, la simplicidad del modelo fue efectiva para los tiempos de entrenamiento y las pruebas unitarias. No obstante, sí hay un tiempo agregado de varios segundos al inicio de la simulación que para el GeNeSys, con muchos individuos y varios ANFIS en cada uno, podría incrementar el tiempo de procesamiento.

5.2. Próximos pasos

Mencionados previamente, se identificaron varios aspectos que pueden ser objeto de mejora en trabajos futuros:

- Desarrollar un modelo de aprendizaje profundo especializado en NLG que tenga propiedades adaptativas que sea capaz de verbalizar los comportamientos particulares de los ANFIS.
- Desarrollar un modelo de comprensión de lenguaje natural (NLU, por sus siglas en inglés, *Natural Language Understanding*) que a partir de los patrones lingüísticos utilizados en NLG, pueda generar valores P, D, L y R, es decir, un proceso inverso al abordado en este trabajo. Con el fin que los robots aprendices tengan a su disposición los valores numéricos necesarios para el aprendizaje cultural.
- Recopilar muestras de un mayor número de individuos del sistema GeNeSys, lo que permitiría identificar patrones que faciliten la generación de lenguaje natural en el proceso de aprendizaje.

Bibliografía

[1] Ratish Puduppully, Yao Fu y Mirella Lapata. *Data-to-text Generation with Variational Sequential Planning*. 2022. arXiv: 2202.13756 [cs.CL].

[2] *Data2Text: Automated Text Generation from Structured Data*. https://www.microsoft.com/en-us/research/project/data2text-automated-text-generation-from-structured-data/.

[3] Martin Hellmann. «Fuzzy Logic Introduction». En: (ene. de 2001).

[4] L.A. Zadeh. «Fuzzy sets». En: *Information and Control* 8.3 (1965), págs. 338-353. ISSN: 0019-9958. DOI: https://doi.org/10.1016/S0019-9958(65)90241-X. URL: https://www.sciencedirect.com/science/article/pii/S001999586590241X.

[5] Dante Sterpin-Buitrago. *GeNeSys - sistema de co-evolución genética y neuro-memética para la auto-organización senso-motriz y conductual en una sociedad de robots*. 2019. URL: http://hdl.handle.net/10554/46082.

[6] Xiexiong Lin et al. *A Logic Aware Neural Generation Method for Explainable Data-to-text*. Ago. de 2022. DOI: 10.1145/3534678.3539082.

[7] Andrea Cascallar-Fuentes et al. «Automatic generation of textual descriptions in data-to-text systems using a fuzzy temporal ontology: Application in air quality index data series». En: *Applied Soft Computing* 119 (2022), pág. 108612. ISSN: 1568-4946. DOI: https://doi.org/10.1016/j.asoc.2022.108612. URL: https://www.sciencedirect.com/science/article/pii/S1568494622001120.

[8] Sam Wiseman, Stuart M. Shieber y Alexander M. Rush. *Challenges in Data-to-Document Generation*. 2017. arXiv: 1707.08052 [cs.CL].

[9] Feng Nie et al. *Operations Guided Neural Networks for High Fidelity Data-To-Text Generation*. 2018. arXiv: 1809.02735 [cs.CL].

[10] Ratish Puduppully, Li Dong y Mirella Lapata. «Data-to-Text Generation with Content Selection and Planning». En: *Proceedings of the AAAI Conference on Artificial Intelligence* 33.01 (2019), págs. 6908-6915. DOI: 10.1609/aaai.v33i01.33016908. URL: https://ojs.aaai.org/index.php/AAAI/article/view/4668.

[11] Samuel Diciembre Sanahuja. «Sistemas de Control con Lógica Difusa: Métodos de Mamdani y de Takagi-Sugeno-Kang (TSK)». En: (2017).

[12] Juan Carlos Gómez. «Fuzzy Control». En: *Buenos Aires: edUTecNe-Editorial Universitaria de la Universidad Tecnológica Nacional* (2008).

[13] Fernando Ornelas Tellez y Michoacan Morelia. «Control con Lógica Difusa». En: *Teoría Conjuntos Difusos. Morelia, Michoacán, México* ().

[14] Carlos A Díaz-Contreras, Alejandra Aguilera-Rojas y Nathaly Guillén-Barrientos. «Lógica difusa vs. modelo de regresión múltiple para la selección de personal». En: *Ingeniare. Revista chilena de ingeniería* 22.4 (2014), págs. 547-559.

[15] R. Jager, H.B. Verbruggen y P.M. Bruijn. «The Role of Defuzzification Methods in the Application of Fuzzy Control». En: *IFAC Proceedings Volumes* 25.6 (1992). IFAC Symposium on Intelligent Components and

Instruments for Control Applications (SICICA'92), Malaga, Spain, 20-22 May 1992, págs. 75-80. ISSN: 1474-6670. DOI: https://doi.org/10.1016/S1474-6670(17)50883-6. URL: https://www.sciencedirect.com/science/article/pii/S1474667017508836.

[16] Robert Aunger. *The electric meme: A new theory of how we think*. Simon y Schuster, 2002.

[17] Tayná Lorena Ramos Gastaldi. *Aplicación de los mapas auto-organizados (SOM) en identificación del valor del cliente para facilitar la selección de los más valiosos*. Inf. téc. Universidad del Valle / Cali, Colombia, 2014. URL: https://bibliotecadigital.univalle.edu.co/handle/10893/17413.

[18] Sepp Hochreiter y Jürgen Schmidhuber. «Long Short-term Memory». En: *Neural computation* 9 (dic. de 1997), págs. 1735-80. DOI: 10.1162/neco.1997.9.8.1735.

[19] Kyunghyun Cho et al. *Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation*. 2014. arXiv: 1406.1078 [cs.CL].

[20] Krzysztof Zarzycki y Maciej Ławryńczuk. «Advanced predictive control for GRU and LSTM networks». En: *Information Sciences* 616 (2022), págs. 229-254. ISSN: 0020-0255. DOI: https://doi.org/10.1016/j.ins.2022.10.078. URL: https://www.sciencedirect.com/science/article/pii/S0020025522011987.

[21] Ilya Sutskever, Oriol Vinyals y Quoc V. Le. *Sequence to Sequence Learning with Neural Networks*. 2014. arXiv: 1409.3215 [cs.CL].

[22] Junyoung Chung et al. *Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling*. 2014. arXiv: 1412.3555 [cs.NE].

[23] DANIEL HERNANDEZ MENDIETA et al. «Sistema Experto Difuso aplicado al diagnóstico de fallas industriales mediante el conocimiento experto obtenido de la herramienta RCM». En: (2016).

[24] Sriparna Majumdar y Aaron Brick. *Recognizing Handwriting Styles in a Historical Scanned Document Using Scikit-Fuzzy c-means Clustering*. 2022. arXiv: 2210.16780 [cs.CV].

[25] Cyberbotics. *Introduction to Webots*. Online. 2023. URL: https://cyberbotics.com/doc/guide/introduction-to-webots.

[26] David M Powers. «Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation». En: *arXiv preprint arXiv:1207.4683* (2007).

[27] Geoffrey Hinton. «Neural networks for machine learning». En: *Coursera* (2012).