INTEGRACIÓN DEL APRENDIZAJE POR REFUERZO EN PARSERS SEMÁNTICOS PARA LA DEDUCCIÓN LÓGICA EN LENGUAJE NATURAL

MAGISTER EN MATEMÁTICAS APLICADAS Y CIENCIAS DE LA COMPUTACIÓN

Camilo Andrés Gómez Vargas

Dirección:

Edgar Jose Andrade

Universidad del Rosario

Escuela de Ingeniería, Ciencia y Tecnología

Maestría en matemáticas aplicadas y ciencias de la computación

AGRADECIMIENTOS

Agradezco especialmente a mi director, Edgar Andrade, por su guía constante, su compromiso académico y su confianza. Edgar no ha sido solamente un director de tesis, sino también una fuente de inspiración a lo largo de mi formación académica, profesional y personal. Su pasión, rigurosidad y entrega me han impulsado a crecer como profesional y profundizar con entusiasmo en distintos temas de interés. Su orientación experta y dedicación incansable han sido un pilar fundamental en este proceso formativo.

También, quiero expresar un agradecimiento muy especial a mi hermana, Kelly Gómez, por ayudarme desde mis primeros pasos en el colegio hasta este momento. Su apoyo ha sido constante y multifacético: desde orientarme en decisiones profesionales y personales, hasta ayudarme a resolver dudas y obstáculos académicos desde su capacidad de analisis y resolución de problemas. Agradezco especialmente por siempre brindarme un respaldo incondicional y único. Su ejemplo, sabiduría y generosidad han sido una guía silenciosa en cada etapa de mi vida.

ABSTRACT

El procesamiento del lenguaje natural (NLP) es una subdisciplina de la inteligencia artificial centrada en la interacción entre las computadoras y los seres humanos mediante lenguaje natural. Su objetivo es desarrollar modelos y sistemas que comprendan, interpreten y generen lenguaje natural de manera similar a como lo hace una persona. En este contexto, los parsers semánticos juegan un papel fundamental, ya que son herramientas que descomponen y representan la estructura y el significado de las oraciones. Estos permiten transformar el texto en una representación formal, proporcionando un medio para que los sistemas inteligentes cuenten con una representación del significado subyacente de las palabras y relaciones. Aunque los parsers semánticos son herramientas de gran importancia en NLP, estos métodos tienden a depender de reglas preestablecidas o de modelos supervisados que aprenden de ejemplos etiquetados, limitando su capacidad para la generalización y representación de nuevas estructuras. Esta falta de flexibilidad de los parsers para adaptarse a nuevas oraciones o a estructuras más complejas sin la necesidad de re-entrenamiento o de una definición más amplia de reglas gramaticales, restringen su utilidad en tareas complejas de inferencia y razonamiento lógico. Por tanto, el objeto de estudio de este trabajo es desarrollar un sistema que utilice aprendizaje por refuerzo profundo para optimizar la representación de estructuras lógicas a partir de oraciones en lenguaje natural. Así, esta investigación desarrolla un modelo capaz de realizar representaciones de silogismos con estructuras conjuntivas e implicatorias. El trabajo se centra en la definición del entorno de aprendizaje, la señal de recompensas, el esquema de entrenamiento y la evaluación de resultados. De esta manera, se busca mejorar la capacidad de las máquinas para interpretar y razonar sobre el lenguaje, lo cual representa un avance en el desarrollo de sistemas de inteligencia artificial que puedan operar con un razonamiento estructurado, consistente y fundamentado.

Natural language processing (NLP) is an artificial intelligence subdiscipline focused on the interaction between computers and humans through natural language. The goal of NLP is to develop models and systems that can un-

derstand, interpret, and generate natural language like a human. Semantic parsers play a fundamental role in this context because they break down and represent the structure and meaning of sentences. Semantic parsers transform text into a formal representation, providing intelligent systems with a means to understand the underlying meaning of words and relationships. However, these methods tend to rely on pre-established rules or supervised models that learn from labeled examples, which limits their ability to generalize and represent new structures. The inability of parsers to adapt to new sentences or more complex structures without retraining or a broader definition of grammatical rules restricts their usefulness in complex inference and logical reasoning tasks. Thus, this study aims to develop a system that uses deep reinforcement learning to optimize the representation of logical structures from sentences in natural language. This research develops a model capable of representing syllogisms with conjunctive and implicative structures. The study focuses on defining the learning environment, reward signal, training scheme, and evaluation of results. This approach aims to enhance machines' ability to interpret and reason about language, representing an advancement in the development of artificial intelligence systems capable of operating with structured, consistent, and well-founded reasoning.

Índice

1. INTRODUCCIÓN **1**

2. OBJETIVOS **4**

2.1. Objetivo general 4

2.2. Objetivos específicos 4

3. PROBLEMA Y JUSTIFICACIÓN **5**

4. MARCO TEÓRICO Y ESTADO DEL ARTE **7**

4.1. Lógica de primer orden 7

4.2. Teoría de representación de discursos 7

4.3. Procesamiento de lenguaje natural 10

4.4. Parsers semánticos 10

4.5. Transformers 12

4.6. Aprendizaje por refuerzo 14

4.7. Aprendizaje por refuerzo profundo 16

4.8. Parsing semántico con aprendizaje por refuerzo profundo 17

5. METODOLOGÍA **19**

5.1. Diseño del modelo de aprendizaje por refuerzo profundo 19

5.2. Datos seleccionados 19

5.3. Experimentación 20

5.4. Evaluación de resultados 20

5.5. Ajuste y mejoramiento del modelo 20

6. ENTORNO DE APRENDIZAJE POR REFUERZO **21**

6.1. Estructura del entorno 21

6.1.1. Espacio de observación 21

6.1.2. Espacio de acciones 22

6.1.3. Función de recompensa 23

6.1.4. Implementación técnica del Entorno 28

6.2. Diseño y entrenamiento del agente 28

6.2.1. Conjunto de datos 28
6.2.2. Aprendizaje curricular 29
6.2.3. Proceso y técnicas de entrenamiento 31
6.2.4. Deep Q-Network 33

7. RESULTADOS Y DISCUSIÓN **34**
7.1. Protocolo de entrenamiento 34
7.2. Capacidades de generalización 38
7.2.1. Análisis cuantitativo 38
7.2.2. Análisis cualitativo 39
7.3. Discusión 42

8. CONCLUSIONES **44**

9. REFERENCIAS **45**

Lista de tablas

1. Parámetros del esquema de entrenamiento para la frase “Some cats are pets” 34
2. Parámetros del esquema de Entrenamiento para las frases “All cats are mammals” y “Some cats are pets” 35
3. Parámetros del esquema de Entrenamiento para la inclusión de nuevos ejemplos. 37
4. Porcentaje de éxito de cada modelo en el conjunto de prueba. El entrenamiento corresponde con el número de frases consideradas dentro del modelo bajo la metodología defina en las tablas 1 y 2 en los casos 1 y 2 y bajo la metodología descrita en la tabla 3 para los demás casos. El porcentaje de $16,6\,\%$ indica que el modelo fue capaz de representar exitosamente 1 de las 6 frases consideradas 38

Lista de figuras

2. Parser semántico basado en RL: el agente procesa una palabra a la vez, generando una DRS (parcial). Los componentes del estado se codifican en un *embedding* utilizando un *transformer auto-encoding*. Una vez finalizado el proceso, la DRS se utiliza para tareas posteriores de respuesta a preguntas. Fuente: Reproducido de [1]. 18
3. Ejemplo del procesamiento bajo las acciones (simplificadas) del entorno para la generación de la DRS para la frase "Some cats are pets". Fuente: Autoría propia. 24
4. Ejemplo del procesamiento bajo las acciones (simplificadas) del entorno para la generación de la DRS para la frase "All cats are mammals". Fuente: Autoría propia. 25
5. Diagrama de flujo de la función de recompensa. Fuente: Autoría propia. 27
6. Estrategia utilizada para el aprendizaje curricular del modelo DRL, aplicada según el marco general del CL. Fuente: Autoría propia. 31
7. Resultados del esquema de entrenamiento definido en la tabla 1. Gráfica de la recompensa recibida por episodio. Fuente: Autoría propia. 35
8. Resultados del esquema de entrenamiento definido en la tabla 2. Gráfica de la recompensa recibida por episodio. Fuente: Autoría propia. 36
9. Recompensa promedio por ventana de 1 000 episodio para el modelo con 5 frases, entrenado según el esquema de la tabla 4. Fuente: Autoría propia. 39
10. Representación en 2 dimensiones de los embeddings que genera cada estado dada la secuencia de acciones necesarias para la representación de frases de tipo "All x are y". Fuente: Autoría propia. 40
11. Representación en 2 dimensiones de los embeddings que genera cada estado dada la secuencia de acciones necesarias para la representación de frases de tipo "Some x are y". Fuente: Autoría propia. 41
12. Embeddings some y all. Fuente: Autoría propia 42

1 INTRODUCCIÓN

El Procesamiento de Lenguaje Natural (NLP, por sus siglas en inglés) ha experimentado un crecimiento exponencial, impulsado por avances en el aprendizaje profundo y la disponibilidad de grandes corpus de texto. Los Modelos de Lenguaje Grandes (LLM, por sus siglas en inglés) son sistemas de inteligencia artificial avanzados que pueden comprender y generar lenguaje natural. Estos modelos han demostrado sorprendentes capacidades en tareas como la traducción automática, el resumen de textos, la clasificación de sentimientos, la respuesta a preguntas, la generación de textos, entre otros [2–5].

Sin embargo, estos modelos funcionan predominantemente sobre asociaciones estadísticas, no sobre una compresión semántica o lógica del contenido. Es por ello que los modelos actuales tienen limitaciones fundamentales. Ellos no consideran una representación explicita de la estructura lógica de un discurso y por esa razón no siguen una secuencia lógica formal sobre hechos, consecuencias o contradicciones. Esto los hace propensos a errores de inferencia o respuestas incoherentes cuando la tarea requiere de deducción [6–8]. Realizar deducciones precisas es esencial en áreas como el derecho, los protocolos médicos y las deducciones matemáticas formales y es aquí donde los LLMs aún tienen muchas oportunidades de mejora.

La representación lógica de discursos permite traducir textos en lenguaje natural a estructuras formales, como la lógica de primer orden. Estas representaciones proporcionan un marco en el que un agente puede realizar tareas como razonamiento multi etapa, verificación de contradicciones o falacias, establecimiento de relaciones lógicas, entre otros [9, 10]. Estos enfoques permiten interpretar enunciados considerando su estructura semántica y relaciones lógicas, proporcionando una base formal, interpretable y verificable para entender cómo un agente toma decisiones o llega a una conclusión especifica.

Sin embargo, a pesar del valor de los métodos de representación mediante formalismos lógicos, su utilización generalizada enfrenta limitaciones significativas. Por un lado, estos métodos tienen una gran complejidad técnica inherente a su comprensión e implementación. Por otro lado, se enfrentan a la dificultad añadida del proceso automático de conversión del lenguaje natural a representaciones lógicas [11, 12]. Aunque existen diversas aproximaciones para automatizar este proceso de conversión, los modelos de redes neuronales profundas (DNN, por sus siglas en inglés), especialmente los

basados en arquitecturas de tipo transformer, han revolucionado el panorama. Estos modelos permiten codificar el lenguaje natural en vectores (embeddings), creando espacios semánticos que favorecen la generalización mediante la proximidad vectorial [3]. No obstante, a pesar de sus ventajas computacionales, la representación vectorial generada de los modelos DNN presentan dificultades intrínsecas para realizar tareas de inferencias lógicas fiables y explicables. Por lo cual, en contextos que requieren procesos inferenciales robustos y transparentes, los métodos clásicos continúan siendo fundamentales, especialmente considerando que las DNN dependen significativamente de grandes cantidades de datos anotados para su entrenamiento.

El propósito de esta investigación es diseñar un parser semántico que sea más eficiente y flexible que los enfoques convencionales. Para ello, se propone el uso de técnicas de aprendizaje por refuerzo profundo, mediante las cuales un agente aprenda a asignar significados semánticos a cada elemento del texto y, a partir de una señal de recompensa, genere sus propias reglas de representación con el fin de optimizar su desempeño.

Se presentan los resultados preliminares que evidencian que el parser semántico propuesto, es capaz de construir representaciones lógicas conjuntivas e implicatorias. El modelo desarrollado tiene indicios de generalización a ejemplos no vistos durante el entrenamiento. Lo que sugiere que el modelo cuenta con buenas capacidades de abstracción de las relaciones semánticas presentes en la representación de cada estado. Estos hallazgos respaldan la viabilidad del enfoque y abren nuevas posibilidades para su extensión a formas lingüísticas más complejas, manteniendo la coherencia lógica en la representación y evaluación de discursos. El trabajo futuro se centrará en extender el conjunto de estructuras lingüísticas que el agente es capaz de representar. También, se desea aumentar las capacidades de generalización mediante la optimización del coste computacional debido a la complejidad inherente de la tarea de representación semántica.

En las secciones que siguen, se introducen los conceptos teóricos necesarios para la comprensión a cabalidad de la implementación desarrollada. Esto incluye los fundamentos de la representación semántica y el aprendizaje por refuerzo. Asimismo, se presenta el trabajo previo que sirve como base para esta investigación, destacando sus aportes y oportunidades. Posteriormente, se describen en detalle la estructura del entorno, el diseño del sistema de recompensas y el proceso de entrenamiento del agente. Finalmente,

se discuten y analizan los resultados obtenidos, con el objetivo de evaluar la viabilidad del enfoque propuesto y sus implicaciones para futuras extensiones del modelo.

2 OBJETIVOS

2.1. Objetivo general

Diseñar e implementar un parser semántico basado en aprendizaje por refuerzo, que permita representar oraciones en lenguaje natural como fórmulas de la lógica de primer orden y evaluar la deducción sobre estas representaciones.

2.2. Objetivos específicos

1. Identificar e implementar en python las reglas de representación dadas por la teoría de representación de discursos extendiendo la representación existente de sujeto acción a oraciones silogísticas.

2. Implementar el sistema de recompensas del parser como un problema de aprendizaje por refuerzo extendiendo las reglas de representación y adaptando la función de recompensas a problemas de implementación lógica más compleja.

3. Analizar y filtrar el conjunto de datos FOLIO entendiendo bien el sistema de representación usado, incluyendo las claves de representación y la longitud de las premisas.

4. Entrenar el sistema de aprendizaje por refuerzo sobre un subconjunto delimitado de FOLIO.

5. Analizar y presentar los resultados.

3 PROBLEMA Y JUSTIFICACIÓN

El razonamiento, entendido como la obtención de nueva información a partir de información dada, es una actividad ubicua de la cognición humana. El razonamiento está presente en procesos como la percepción, comprensión de textos, interpretación de situaciones sociales, entre otros. Existen dominios específicos donde el razonamiento tiene una relevancia crucial, ya que requiere llevar a cabo una cadena extensa de pasos lógicos donde cada uno de ellos debe ser correcto. Algunos ejemplos de estos dominios incluyen las matemáticas, las leyes, los protocolos médicos, el arbitraje de regulaciones e incluso los juegos. En estos contextos, no solo es esencial contar con una cadena explícita de justificación, sino que cada paso de la cadena debe ser 100 % correcto.

La información propia de estos dominios no existe (o no se maneja) en un medio de representación formal, lo que representa un problema, ya que no permite el uso de herramientas lógicas y, menos aún, que sea automatizable. Para atender la formalización de este tipo de información, dentro del NLP se disponen los parsers semánticos, esto es, sistemas diseñados bajo reglas gramaticales y semánticas que asignan un significado a un texto mediante el análisis de su estructura, atribuyendo una interpretación a cada uno de sus elementos [13]. Este enfoque permite un análisis profundo del texto para lograr una comprensión precisa, al facilitar la transformación del lenguaje en representaciones estructuradas y formales. Sin embargo, estos métodos dependen en gran medida de reglas preestablecidas y de datos etiquetados [14]. Esta aproximación limita la capacidad de estos sistemas para generalizar nuevas estructuras o representar significados complejos de manera robusta, lo que restringe su utilidad en tareas complejas de inferencia y razonamiento lógico, donde es esencial generar inferencias de precisión absoluta a partir de estructuras formales.

En este contexto, los avances en inteligencia artificial y NLP ofrecen alternativas para el procesamiento de dicho tipo de información. Los LLMs han demostrado capacidades a diferentes niveles de precisión en tareas como la generación de textos coherentes y responder preguntas basándose en el conocimiento almacenado (patrones aprendidos de grandes volúmenes de datos) [15]. Estas arquitecturas, basadas en transformers [2], permiten identificar patrones en el lenguaje mediante la vectorización de palabras y la predicción contextual. No obstante, enfrentan limitaciones críticas en tareas que requieren razonamiento lógico y formal [16]. Un problema clave es el fenómeno conocido

como “alucinaciones”, que se refiere a la generación de respuestas que se ven plausibles, pero carecen de respaldo factual o evidencia verificable. Este fenómeno ocurre porque el modelo se enfoca en maximizar la probabilidad de la siguiente palabra en función de patrones estadísticos, sin evaluar la veracidad o la evidencia de las respuestas. Esta carencia también afecta su capacidad de razonamiento lógico, ya que no poseen la habilidad de desarrollar múltiples pasos en una cadena lógica formal, impidiendo resolver problemas de manera detallada o producir cadenas lógicas extensas y precisas [6,17,18].

De esta manera, aunque los modelos y sistemas existentes de IA han demostrado un nivel satisfactorio de rendimiento en diferentes campos, su capacidad para abordar problemas que requieren de razonamiento lógico formal sigue siendo limitada, lo cual subraya la necesidad de explorar enfoques más robustos. La integración del aprendizaje por refuerzo en el diseño y desarrollo de un parser semántico supone una ventaja al permitir la representación y deducción de significados semánticos sin la necesidad de una estructura rígida de reglas predefinidas. Con el presente trabajo, se espera desarrollar un modelo de aprendizaje por refuerzo profundo que permita esquemas de razonamiento lógico consistentes, contribuyendo a la integración y confiabilidad entre los modelos de lenguaje natural y sus diferentes aplicaciones y usos en campos de rigurosidad lógica y deductiva.

4 MARCO TEÓRICO Y ESTADO DEL ARTE

4.1. Lógica de primer orden

La lógica es la disciplina que, dentro de su amplio ámbito, estudia la inferencia, es decir, el proceso mediante el cual se extraen conclusiones a partir de un conjunto de premisas. En este contexto, la validez de una inferencia depende de que la conclusión esté lógicamente relacionada con las premisas, de manera que la veracidad de estas se transfiera a la conclusión. La relación entre las premisas y la conclusión que garantiza la transferencia de verdad es una relación formal, lo que significa que depende exclusivamente de la estructura lógica de los enunciados y no de su contenido específico [9,10].

En términos generales, la lógica deductiva, proporciona un marco formal que permite deducir conclusiones de un conjunto de premisas utilizando reglas de inferencia bien definidas. Dentro de este marco, la lógica de primer orden amplía las capacidades de la lógica proposicional al permitir la formulación de enunciados que expresan: (i) individuos; (ii) predicados y relaciones entre individuos; (iii) variables de individuo y cuantificación sobre ellos. Esto permite representar de manera más precisa estructuras complejas de conocimiento.

Los dos cuantificadores esenciales en lógica de primer orden son:

- **Cuantificador universal** ($\forall$): indica que una proposición es verdadera para todos los elementos de un conjunto.
- **Cuantificador existencial** ($\exists$): afirma que una proposición es verdadera para al menos un elemento del conjunto.

El uso de estos cuantificadores, en combinación con predicados, permite construir enunciados detallados sobre las relaciones entre los elementos del dominio. Esto resulta esencial en áreas como el procesamiento de lenguaje natural y la teoría de representación de discursos, donde la capacidad de expresar relaciones y generalizaciones dentro de un conjunto de datos es crucial para la inferencia y el razonamiento automatizado [9].

4.2. Teoría de representación de discursos

La teoría de representación de discursos (DRT, por sus siglas en inglés) es un marco teórico desarrollado para modelar y comprender la interpretación y representación de discursos o textos. Uno de los problemas centrales que busca resolver es el paradigma

de la anáfora [9]. Este problema radica en la representación de los vínculos entre los pronombres y sus antecedentes a través de los límites de la frase. Para dar solución a ello, esta teoría introduce las estructuras de representación del discurso (DRS, por sus siglas en inglés), en las cuales se interpretan las frases sustantivas como introductoras de referentes en el discurso y de esta forma, se definen como variables libres de uso para anáforas en el discurso posterior representando así grandes unidades discursivas.

Las DRSs están compuestas por 2 principales elementos: una lista de referentes y una lista de condiciones. Comúnmente, estos elementos son representados en un cuadro estructurado donde en la parte superior se listan los referentes y en la parte inferior las condiciones (ver Figura 1). Los referentes son variables libres que representan entidades en el discurso y las condiciones pueden ser proposiciones atómicas, conectores o condiciones complejas con DRSs [9, 19]. Las proposiciones atómicas corresponden a nombres de predicados aplicados a diferentes referentes del discurso, por ejemplo JUAN(x) es una proposición atómica en la que JUAN es un predicado aplicado al referente del discurso x. Los conectores son expresiones de igualdad entre dos referentes del discurso, como $x = y$ en la Figura 1b. Finalmente, las condiciones complejas corresponden (i) a la negación de una DRS; y (ii) a la implicación entre dos DRSs, como en la Figura 1c.

El procesamiento de las estructuras de representación se realiza oración por oración de manera iterativa. Así, dado un discurso $D = S_1, S_2, \ldots, S_n$ y una DRS vacía K_0, se procesa cada oración S_i, con $i \in \{1, \ldots, n\}$, de modo que el análisis semántico sintáctico de S_i —es decir, la extracción de referentes y condiciones de la frase— se añade a la DRS K_{i-1} aplicando los principios de construcción (ver Figura 1b). Esto implica simplificar las condiciones hasta obtener formas irreducibles [9, 19]. De esta forma, cada oración procesada, aporta información a la DRS construida teniendo en cuenta las oraciones anteriores. De manera que, la construcción de una DRS permite identificar y enlazar pronombres o expresiones referenciales con sus antecedentes en un discurso.

La DRT formaliza estas relaciones para garantizar una interpretación coherente y lógica de todo el discurso. Por ejemplo, en la frase "Juan encontró un gato. Él lo adoptó", el pronombre "él" se refiere a "Juan", y "lo" al "gato". En la figura 1a se puede ver la representación de la primera frase y en la figura 1b se puede observar como se integra la información de la segunda frase a la DRS inicial aplicando los principios de construcción. Este ejemplo evidencia el enlace entre referentes en un discurso y su

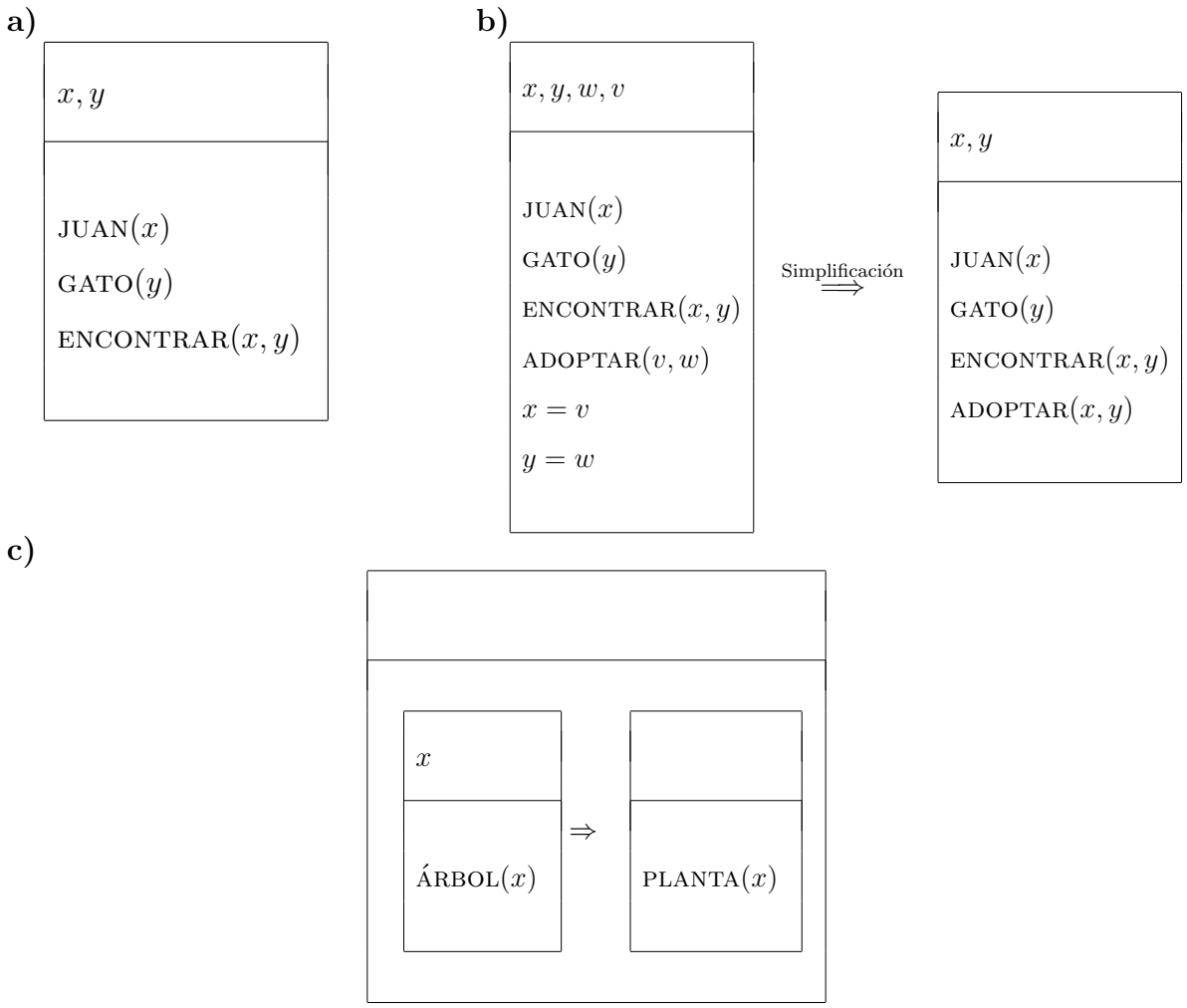

Figura 1: Ejemplos de estructuras de representación de discursos. (a) DRS apropiada para la oración "Juan encontró un gato". (b) DRS inclusión del procesamiento de la oración "Él lo adoptó" en la DRS a. (c) DRS apropiada para la oración "Todo árbol es una planta". Fuente: Autoría propia.

resolución bajo este enfoque.

En el campo de la lingüística computacional, la DRT tiene una amplia aplicación. Principalmente, es aplicada en sistemas basados en inteligencia artificial para la resolución de anáforas gracias a su eficiencia para abordar las complejidades del discurso en lenguaje natural [19].

4.3. Procesamiento de lenguaje natural

El procesamiento de lenguaje natural (NLP, por sus siglas en inglés) es el conjunto de métodos que hace que el lenguaje humano sea accesible a los computadores [4]. El NLP abarca una amplia gama de tareas, métodos y fenómenos lingüísticos entre los cuales, de manera general, se pueden destacar las siguientes cuatro técnicas más utilizadas:

- Tokenización: División del texto en unidades más pequeñas, como palabras o subpalabras.
- Etiquetado: Asignación de etiquetas a las palabras para identificar su función gramatical.
- Análisis sintáctico: Identificación de la estructura gramatical de las oraciones.
- Análisis semántico: Entendimiento del significado de las palabras y las relaciones entre ellas.

4.4. Parsers semánticos

Uno de los temas mas relevantes de investigación dentro del NLP, ha sido la comprensión del lenguaje natural. Este campo se centra en la extracción del significado, la intención y el contexto a partir de entradas de lenguaje natural. A raíz de ello, los parsers semánticos surgen como interpretes que buscan establecer una correspondencia entre entrada de lenguaje natural y representaciones de significado. Estas representaciones de significado deben ser estructuras que permitan una comprensión profunda del texto y que puedan ser utilizadas en tareas tales como la comprensión y ejecución de órdenes para la navegación robótica, la exploración y el análisis de datos mediante el análisis sintáctico del lenguaje natural para consultas a bases de datos y el análisis sintáctico de consultas en agentes conversacionales [20].

Los primeros modelos de parsing semántico se basaron en reglas elaboradas a mano para convertir el lenguaje natural en formas lógicas. Estos modelos se limitaban a ámbitos muy restringidos debido al gran esfuerzo manual necesario para crear y mantener las reglas [20]. Posteriormente, se exploraron diferentes métodos principalmente basados en técnicas de aprendizaje estadístico. Existen modelos que integran los sistemas de reglas con técnicas estadísticas, donde utilizan modelos entrenados en pares de frases y sus correspondientes formas lógicas. Con ello, haciendo uso de una gramática categórica combinatoria probabilística (Probabilistic CCG), estos modelos buscan clasificar los posibles análisis sintácticos en función de su probabilidad [21]. Otras aproximaciones utilizan la supervisión débil (weak supervision), en las cuales el modelo es entrenado utilizando datos en los que sólo se proporciona la denotación final (la respuesta), lo que reduce la necesidad de anotaciones detalladas y permite un entrenamiento en un conjunto de datos más amplio [20]. Enfoques más recientes, como los sistemas de codificación y descodificación (Encoder-Decoder Frameworks) o las máquinas simbólicas neuronales (Neural Symbolic Machines), están basados en redes neuronales. Los sistemas Encoder-Decoder son sistemas que codifican una entrada de longitud variable en un vector de tamaño fijo y luego lo decodifican en su forma lógica. Por otro lado, las máquinas simbólicas neuronales, combinan las redes neuronales con razonamiento simbólico, donde el objetivo del modelo es aprender a generar formas lógicas. Para ello, comúnmente se utilizan redes neuronales con memoria aumentada [22].

En el contexto de la representación en lógica de primer orden, una aproximación destacada es el marco propuesto por Lu et al. [23] quienes modelaron este problema como una tarea de traducción de secuencia a secuencia. Para ello, desarrollaron un modelo de aprendizaje por refuerzo dual en el que, de manera conjunta, un componente es entrenado para generar expresiones lógicas; y otro para reconstruir oraciones en lenguaje natural. Durante el entrenamiento, definieron un sistema de recompensas que permite aprender automáticamente una función de validez. Esta función busca evaluar la coherencia semántica entre las frases en lenguaje natural y sus correspondientes expresiones lógicas. El modelo propuesto permite superar las limitaciones de recompensas basadas en reglas manuales predefinidas o restringidas a contextos específicos [23], sin embargo, requiere de un gran volumen de datos etiquetados para lograr una abstracción de los componentes semánticos con un cierto grado de precisión.

Independientemente de las aproximaciones o modelos desarrollados en este campo, persisten retos fundamentales como la dependencia de reglas predefinidas, la complejidad en la representación y la curva de aprendizaje, la necesidad de datos etiquetados, y la validación manual de las representaciones generadas.

4.5. Transformers

Los transformers, son una arquitectura del campo del aprendizaje profundo que dieron base a los grandes modelos de lenguaje (LLMs, por sus siglas en ingles). Este tipo de arquitectura se basa exclusivamente en mecanismos de atención, eliminando la necesidad de capas recurrentes o convolucionales. La innovación en esta arquitectura es el modelo de atención utilizado, el cual hace uso de la "atención multi-cabeza" (multi-head attention), que permite al modelo enfocarse en diferentes posiciones de la secuencia de entrada simultáneamente. Este método permite capturar dependencias a largo plazo en las secuencias de entrada, dotando al modelo de una "memoria" a largo plazo que le permite tener un mayor contexto en las tareas dispuestas. Los transformers están compuestos por un codificador (encoder) y un decodificador (dencoder), ambos formados por múltiples capas de atención y capas feed-forward completamente conectadas. Cada cabeza de atención opera en un espacio de características diferente, lo que permite al modelo capturar diversas relaciones y patrones de datos [2].

Los modelos con arquitectura basada en transformers han demostrado, a través de diversos experimentos, que las cabezas de atención aprenden a enfocarse en distintos aspectos de la estructura del texto de entrada. Un marco de cuatro clases categoriza las funciones que pueden desempeñar las cabezas de atención en un modelo como: recuperación de conocimientos (KR, por sus siglas en inglés), identificación en contexto (ICI, por sus siglas en inglés), razonamiento latente (LR, por sus siglas en inglés) y preparación de expresiones (EP, por sus siglas en inglés) [24].

Las cabezas KR se enfocan en recuperar información relevante almacenada en los parámetros del modelo, actuando como una forma de memoria interna. Las cabezas ICI se encargan de identificar elementos y relaciones clave dentro del contexto del texto de entrada. Por su parte, las cabezas LR manipulan y combinan la información recuperada con los elementos identificados en el texto, con el objetivo de generar relaciones inferenciales. Finalmente, las cabezas EP preparan y estructuran la respuesta final del modelo, asegurando coherencia lingüística y fidelidad al proceso interno de atención.

Dentro del modelo, las distintas etapas o zonas de atención, junto con las redes neuronales feed-forward presentes en los LLMs, resultan fundamentales tanto para almacenar el conocimiento (contexto) como para facilitar los procesos de razonamiento [24].

Dado el desempeño observado de diferentes LLMs en diferentes tareas a distintos niveles de complejidad, múltiples estudios recientes han enfocado sus esfuerzos en evaluar de forma más rigurosa las capacidades de estos modelos. En cuanto a tareas de razonamiento lógico, Zhou et al. proponen el marco FineLogic, el cual introduce una evaluación en tres dimensiones: precisión global de referencia, solidez por pasos y alineación a nivel de representación [25]. Este enfoque permite ir más allá de la precisión de la respuesta final, proporcionando una visión más detallada sobre la interpretabilidad y coherencia estructural del razonamiento (proceso de inferencia) generado por los modelos. El estudio destaca cómo los diferentes estilos de supervisión afectan el desempeño en razonamiento lógico de los LLMs. En particular, la supervisión en lenguaje natural favorece la generalización, incluso en contextos extensos. El estudio concluye que el afinamiento supervisado (SFT, por sus siglas en inglés) mejora principalmente el comportamiento del razonamiento a través de la generación paso a paso de una respuesta. Esto fortalece la coherencia lógica interna más que la simple predicción de respuestas correctas, reforzando la capacidad de los modelos para producir cadenas de razonamiento lógicas. Sin embargo, están limitados en cuando al detalle en el proceso y limpieza de los datos [25]. En orden con las observaciones realizadas, los modelos entrenados con métodos simbólicos mejoran las capacidades en tareas de razonamiento lógico ya que tienden a generar inferencias más estructuradas y consistentes, mejorando su desempeño estructural pero no su precisión [25–27].

En paralelo, ha surgido un interés creciente por los grandes modelos de razonamiento (LRM, por sus siglas en inglés), diseñados específicamente para potenciar las capacidades de razonamiento mediante procesos detallados. Estudios recientes han revelado hallazgos clave al comparar estos modelos con LLMs estándar bajo supuestos de inferencia equivalentes. Por ejemplo, si bien los LLMs convencionales logran un rendimiento comparable en tareas establecidas como MATH500 y AIME24, los LRMs muestran una ventaja significativa en tareas de mayor complejidad como AIME25, lo cual sugiere que su fortaleza radica en el manejo de problemas complejos o benchmarks menos contaminados por datos previos [28–30].

Este comportamiento diferencial ha sido caracterizado en tres regímenes de desempeño: para tareas de baja complejidad, los LLMs superan a los LRMs; en tareas de complejidad media, los LRMs se benefician de sus procesos de razonamiento adicionales; mientras que, en tareas altamente complejas, ambos modelos colapsan en su rendimiento, revelando limitaciones fundamentales. Además, los LRMs presentan un límite de escalabilidad contraintuitivo: su esfuerzo de razonamiento crece con la complejidad hasta cierto punto, para luego disminuir, lo que representa una limitación intrínseca en el desempeño del razonamiento con respecto a la complejidad del problema [28]. Si bien los LRMs pueden identificar soluciones correctas en problemas simples, exhiben dificultades para sostener ejecuciones lógicas consistentes en escenarios más complejos. En conjunto, estos hallazgos enfatizan tanto el potencial como las limitaciones fundamentales de los modelos actuales, y subrayan la necesidad de investigaciones más profundas sobre sus capacidades reales para el razonamiento lógico complejo [31, 32].

4.6. Aprendizaje por refuerzo

El aprendizaje por refuerzo (RL, por sus siglas en inglés) es una rama del aprendizaje automático (ML, por sus siglas en inglés) que busca que un sistema (agente) aprenda a tomar decisiones en situaciones específicas mediante la interacción con un entorno. El objetivo principal del agente es maximizar la recompensa acumulada a lo largo del tiempo en función de un sistema de recompensas bien definido. A través de este proceso, el agente adapta sus estrategias de decisión y la selección de acciones basándose en los resultados previos [33]. Este enfoque del ML se distingue por tres características principales: (i) las acciones influyen en las entradas posteriores (bucle cerrado / retroalimentación); (ii) no hay instrucciones directas sobre qué acciones debe tomar el sistema; (iii) y las consecuencias de las acciones, incluidas las señales de recompensa, se desarrollan a lo largo de periodos prolongados.

El aprendizaje por refuerzo cuenta con siete componentes claves:

- Entorno: todo aquello con lo que el agente interactúa y que cambia de estado como consecuencia de las acciones del agente.
- Agente: el sistema de aprendizaje que observa el entorno y toma decisiones para actuar sobre este.
- Estado: representación de la situación actual del entorno.

- Acciones: las acciones que puede tomar el agente.
- Recompensa: señal que indica cuan buena o mala fue la acción tomada para un estado determinado.
- Política: estrategia que sigue el agente para tomar una acción en función de un estado determinado.
- Función de valor: mide que tan buena es una acción en el largo plazo. Representa la recompensa que el agente puede lograr acumular en el futuro partiendo de ese estado siguiendo una política dada.

El RL se ha consolidado como una técnica clave y de exploración en la aproximación para diferentes problemas del campo del NLP, ofreciendo soluciones efectivas para tareas complejas que implican decisiones secuenciales. A diferencia del aprendizaje supervisado tradicional, que se basa en ejemplos etiquetados, el RL permite a los agentes aprender mediante la interacción con su entorno, obteniendo retroalimentación en forma de recompensas o penalizaciones [33]. En el contexto del NLP, diferentes aplicaciones muestran beneficios como en tareas de generación de resúmenes, traducción automática, respuesta de preguntas, sistemas conversacionales y análisis semántico sintáctico (parsers) [34–37]. La capacidad del RL para optimizar políticas de acción en entornos dinámicos y su flexibilidad en diversas tareas lingüísticas lo posicionan como una herramienta clave para mejorar la precisión y la fluidez en el procesamiento del lenguaje natural [38].

Uno de los algoritmos más usados es el Q-learning, el cual está basado en la función de valor que utiliza una tabla Q para almacenar las recompensas esperadas para cada estado-acción. El objetivo en este método es encontrar la política óptima que maximice la recompensa acumulada en el futuro [39]. El funcionamiento de este algoritmo consta de un proceso iterativo de tres etapas: (i) primero inicializa la tabla Q con valores arbitrarios; (ii) luego observa el estado actual del entorno y selecciona una acción utilizando la política ϵ-greedy (exploración y explotación); (iii) y por último ejecuta la acción y con la recompensa recibida actualiza la tabla Q utilizando la fórmula de Q-Learning:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left[r + \gamma \cdot \text{máx}(Q(s',a')) - Q(s,a) \right]$$

donde:

- $Q(s, a)$ es el valor de la tabla Q para el estado s y la acción a.
- α es la tasa de aprendizaje.
- r es la recompensa recibida.
- γ es el factor de descuento.
- máx$(Q(s', a'))$ es el máximo valor de la tabla Q para el nuevo estado s' y todas las acciones posibles.

La aplicabilidad de este tipo de métodos se limita a dominios que cuentan con pocas características o con espacios de estado de baja dimensión totalmente observables.

4.7. Aprendizaje por refuerzo profundo

El aprendizaje por refuerzo profundo (DRL, por sus siglas en inglés) combina el RL con redes neuronales profundas (DNN, por sus siglas en inglés) permitiendo a los agentes una interacción en entornos más complejos. Según la técnica implementada, las redes neuronales profundas se emplean para aproximar la función de valor y/o optimizar la política del agente. Este tipo de redes permite manejar espacios de estado y acción significativamente más grandes y complejos, al extraer características directamente de los datos observados. Esta propiedad facilita la generalización del entorno sin necesidad de ingeniería manual [40].

Dentro del DRL, existen diferentes técnicas dependiendo del enfoque, los cuales pueden ser clasificadas en dos grupos: métodos de gradiente (Policy Gradient Methods, en inglés) y los métodos basados en la función de valor (Value-Based Methods). Uno de los algoritmos de mayor popularidad dentro de los métodos basados en la función de valor es el Deep Q-Learning (DQN) [41]. Este algoritmo esta basado en el Q-Learning, pero usando redes neuronales para aproximar la función de valor, lo que le permite aprender políticas exitosas a partir de estados de alta dimensión [41].

El DQN tiene como objetivo aproximar la función Q utilizando una red neuronal profunda. Comúnmente, este algoritmo implementa capas convolucionales y fully connected para extraer características relevantes del estado y calcular el valor Q para cada acción posible. El funcionamiento de este algoritmo sigue los mismos pasos iterativos del Q-Learning con la diferencia que, una vez recibida la recompensa por la acción

ejecutada, se calcula el error entre la predicción de la red y la recompensa real. Posteriormente, se actualizan los pesos de la red bajo un algoritmo de optimización [41]. De esta forma, el objetivo del agente es maximizar la recompensa futura acumulada y la función de acción-valor óptima es aproximada como:

$$Q^*(s,a) = \max_{\pi} E\left[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + ... | s_t = s, a_t = a, \pi\right]$$

4.8. Parsing semántico con aprendizaje por refuerzo profundo

El uso del aprendizaje por refuerzo en tareas de Parsing semántico supone un enfoque que permite la generación satisfactoria de representaciones adecuadas para la realización de inferencias lógicas fiables sin la dependencia de un conjunto de reglas preestablecidas, es decir, permitir la formulación autónoma de las reglas del parser semántico. Una aproximación inicial de este enfoque muestra que, bajo el uso de la DRT, un agente es capaz de generar representaciones satisfactorias en entornos simples. Esto indica que el agente es capaz de aprender las reglas que rigen el proceso de conversión, lo que garantiza la fiabilidad en la inferencia lógica [1].

El sistema desarrollado por [1] (ver Figura 2) emplea un modelo independiente de tipo transformer para generar representaciones vectoriales intermedias que son utilizadas por el agente de DRL para producir una DRS. De esta manera, este método se beneficia de las capacidades de generación y captura de dependencias de los transformers y de las ventajas en tareas de resolución secuencial de problemas del DRL [2, 42]. El entorno implementado, consta de 5 acciones que permiten la generación de representaciones para las estructuras gramaticales correspondientes al sujeto y verbo. El entrenamiento del modelo se desarrolló utilizando un conjunto de datos de 100 frases. Los resultados obtenidos muestran que el agente alcanza una recompensa media de 98 puntos, donde la recompensa máxima es 100 y tiene una tasa de éxito del 100 % en la generación de respuestas. Esto indica que el modelo es capaz de generar satisfactoriamente oraciones que tienen como estructura "nombre propio + verbo" y responder a preguntas de la forma "¿Quién + verbo?" y "¿Qué hace + nombre propio?" [1].

Estos resultados obtenidos dejan en evidencia que el uso del aprendizaje por refuerzo para el diseño de un parser semántico permite la asignación satisfactoria de interpretaciones semánticas en entornos sencillos. Simultáneamente, muestra barreras de rendimiento en la capacidad del agente para diferenciar entre estados debido al

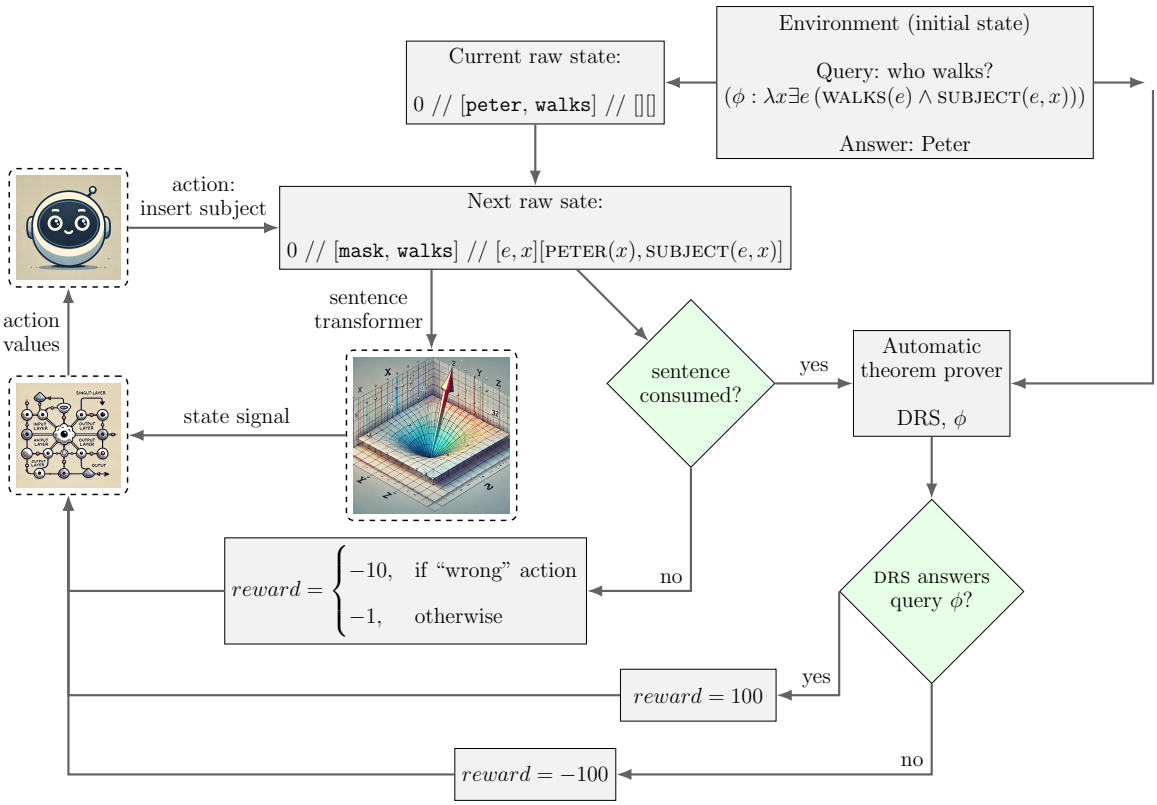

Figura 2: Parser semántico basado en RL: el agente procesa una palabra a la vez, generando una DRS (parcial). Los componentes del estado se codifican en un *embedding* utilizando un *transformer auto-encoding*. Una vez finalizado el proceso, la DRS se utiliza para tareas posteriores de respuesta a preguntas. Fuente: Reproducido de [1].

solapamiento entre los embeddings generados por el modelo transformer, lo cual no representa un impacto en el desempeño satisfactorio del agente pero deja en evidencia uno de los desafíos que enfrenta este enfoque [1].

El presente trabajo realiza una extensión del entorno definido en [1] con el fin de extender la gama de estructuras oracionales que el agente es capaz de representar, centrándose en la extensión del espacio de acciones, las señales de recompensa, la evaluación de resultados y la optimización del comportamiento y rendimiento del agente.

5 METODOLOGÍA

El desarrollo de la presente investigación se llevará a cabo en cinco etapas principales. A continuación, se menciona cada una de estas junto con las actividades involucradas respectivamente para cada etapa.

5.1. Diseño del modelo de aprendizaje por refuerzo profundo

Esta primera etapa comprende la creación del entorno de aprendizaje reforzado. Esto incluye el diseño de la función de recompensas, la definición de las posibles acciones y las transiciones de estados. De igual forma, comprende la selección de la arquitectura de la red neuronal para el modelo DRL y el ajuste de los parámetros del entorno para el aprendizaje del modelo. Los detalles de esta etapa se presentan en la Sección 6.1

5.2. Datos seleccionados

Se utilizará FOLIO, un conjunto de datos de razonamiento en lenguaje natural anotado por humanos, lógicamente complejo y diverso, dotado de anotaciones de lógica de primer orden (FOL) [43]. A continuación, se detallan las tareas enmarcadas en esta etapa:

a. Análisis y estandarización del conjunto de datos. Se validan los ejemplos del conjunto de datos, las anotaciones en lógica de primer orden y la complejidad de las oraciones.

b. Adquisición de los datos de entrenamiento, validación y prueba. Se divide el conjunto de datos en tres subconjuntos de la siguiente manera:

   i. Conjunto de entrenamiento: interacción del agente con el entorno para el aprendizaje de las políticas de acción optimas.

   ii. Conjunto de validación: Medición del rendimiento para ajustar los hiperparámetros del modelo.

   iii. Conjunto de prueba: Medir la calidad de la política aprendida.

Los detalles de esta etapa se presentan en la Sección 6.2.1

5.3. Experimentación

Esta etapa se basa en el entrenamiento, evaluación y optimización del modelo DRL. A continuación, se detallan las tareas enmarcadas en esta etapa:

a. Entrenamiento del modelo haciendo uso del conjunto de entrenamiento.

b. Evaluación del modelo en el conjunto de prueba.

c. Optimización de hiperparámetros y ajuste del modelo en función de los resultados de evaluación.

Los detalles de esta etapa se presentan en las Secciones 6.2.2, 6.2.3 y 6.2.4

5.4. Evaluación de resultados

Comprende la evaluación del modelo utilizando métricas de rendimiento especificas, la comparación con otros enfoques tradicionales de parsers semánticos y el análisis de los tiempos de inferencia y recursos computacionales requeridos.

5.5. Ajuste y mejoramiento del modelo

En esta etapa se identifican los casos o las estructuras en los que el modelo falla en tareas de análisis semántico. Estamos interesados en explorar las capacidades de generalización de manera cuantitativa, usando un conjunto de prueba distinto al del entrenamiento, y de manera cualitativa, observando las similitudes geométricas de los *embeddings* en el espacio vectorial de los estados.

6 ENTORNO DE APRENDIZAJE POR REFUERZO

En esta sección se presenta una visión general del entorno y del agente de aprendizaje por refuerzo, detallando la estructuración del entorno y las técnicas utilizadas para el entrenamiento. Este entorno se basa en el entorno de *parsing* semántico presentado en la Sección 4.8.

6.1. Estructura del entorno

6.1.1. Espacio de observación

El espacio de observación del agente está compuesto por cuatro elementos principales: (i) el estado actual de procesamiento de la frase, (ii) los metadatos relacionados con dicho proceso, (iii) la frase a representar inalterada, y (iv) y la fórmula en FOL generada a partir del estado de procesamiento. En detalle, el componente (i) es el estado de procesamiento se representa como una cadena de caracteres en la que las palabras ya procesadas son reemplazadas por el símbolo MASK, mientras que las palabras aún no procesadas se representan mediante sus lemas correspondientes. El componente (ii) son los metadatos asociados al proceso incluyen el índice de la palabra actualmente en procesamiento (es decir, aquella sobre la cual se aplica la acción) y el nivel de profundidad de la DRS generada. Esto último hace referencia a que, si la DRS contiene varias sub-DRS, cada DRS anidada o contenida incrementa el nivel de profundidad. El componente (iii) es la frase a representar corresponde a la oración original proporcionada por el entorno. Finalmente, el componente (iv) es la representación actual generada a partir del estado de procesamiento corresponde a la fórmula FOL derivada de la DRS principal en su estado actual. Este procedimiento se realiza de tal forma que, si la DRS no está completamente construida y no es posible derivar su representación en FOL, se completa automáticamente con condiciones vacías en los campos faltantes.

El estado crudo del agente está definido como la lista de las 4 cadenas de caracteres generada por los componentes mencionados. Los elementos correspondientes al estado actual de procesamiento, la frase a representar y la representación actual, son procesados mediante el uso del autocodificador del transformer “distilusebase-multilingual-cased-v1” de la librería Sentence Transformers. El tamaño de la salida que proporciona el autocofidicador es de 384 dimensione. Por lo cual, con el procesamiento de estos tres elementos del estado crudo, se obtiene un tensor de tres dimensiones de tamaño 384.

Este tensor es aplanado con el fin de objeter un vector unidimensional de tamaño 1536. Posteriormente a este vector se le concatena los metadatos correspondientes obteniendo un vector de tamaño 1538.

6.1.2. Espacio de acciones

El entorno cuenta con un espacio de acción discreto de dimensión 13. Estas posibles acciones se pueden agrupar en 3 diferentes categorías: navegación, modificación y creación. A continuación, se presenta cada acción definida:

- **Acciones de navegación:** estas acciones permiten al agente explorar y seleccionar diferentes partes del discurso. De igual forma, permite omitir (enmascarar) una palabra en caso de ser necesario.

    1. Mover derecha: avanzar un token en la secuencia del discurso.
    2. Mover izquierda: retroceder un token en la secuencia del discurso.
    3. Enmascarar: marca el token actual en la secuencia del discurso como una palabra ya procesada.

- **Acciones de modificación:** estas acciones permiten agregar contenido dentro del DRS que el agente está construyendo en la simulación. Además, estás acciones son seguidas de enmascarar y mover a la derecha para poder marcar como procesada la palabra actual y disponer el entorno para procesar la siguiente palabra.

    4. Incluir sustantivo: añade un referente de tipo objeto y un predicado de sustantivo a la DRS.
    5. Incluir sustantivo sin referente: añade un predicado de sustantivo a la DRS sin introducir un nuevo referente.
    6. Incluir sustantivo plural: introduce un referente de tipo conjunto (dos referentes) y se asocia con un predicado de sustantivo plural en la DRS, es decir, un predicado de sustantivo para cada referente y una condición de no igualdad entre los referentes introducidos.
    7. incluir igualdad: añade un condición de identidad a la DRS entre los dos últimos referentes añadidos.
    8. Incluir relación: añade un predicado relacional entre los dos últimos referentes añadidos.

9. Incluir constante: introduce una entidad única al DRS asociada al lema en la posición actual del discurso. Esta acción genera una afirmación de existencia y unicidad en la representación en FOL.

- **Acciones de creación:** estas acciones se emplean para generar una nueva subestructura (DRS). La construcción y el manejo de estas subestructuras se realiza mediante un árbol ternario, en el cual la raíz almacena la DRSprincipal y cada hoja representa una DRSde negación, antecedente o consecuente. De esta forma, el nivel de profundidad en el espacio de observación corresponde al nivel del nodo actual en el árbol. Por otro lado, esta estructura permite el manejo de las condiciones complejas en la DRS.

10. Crear DRS antecedente: generar una subestructura correspondiente a la parte antecedente de una expresión condicional
11. Crear DRS consecuente: generar una subestructura correspondiente a la parte consecuente de una expresión condicional
12. Crear DRS negación: generar una subestructura para representar el contenido de una expresión negada, encapsulándola dentro de un operador de negación
13. Subir nivel: Finalizar la construcción de una subestructura y retornar al nivel superior en la estructura discursiva, integrando la representación de la DRS actual en la DRS principal.

Para ejemplificar la aplicación del espacio de acciones, considere las frases "Some cats are pets" y "All cats are mammals". En las figuras 3 y 4 se muestra respectivamente la secuencia de acciones necesarias y el proceso de creación de las DRSs

6.1.3. Función de recompensa

Para poder definir la función de recompensa, se deben tener en cuenta las reglas del entorno para poder establecer la definición de una acción incorrecta. A continuación, se describen las reglas que definen una acción incorrecta en el entorno:

- El agente no puede moverse a la izquierda si el índice en la secuencia del discurso es 0, es decir, si está ubicado en la primera palabra.
- El agente no puede moverse a la derecha si el índice en la secuencia del discurso es el máximo, es decir, si está ubicado en la última palabra.

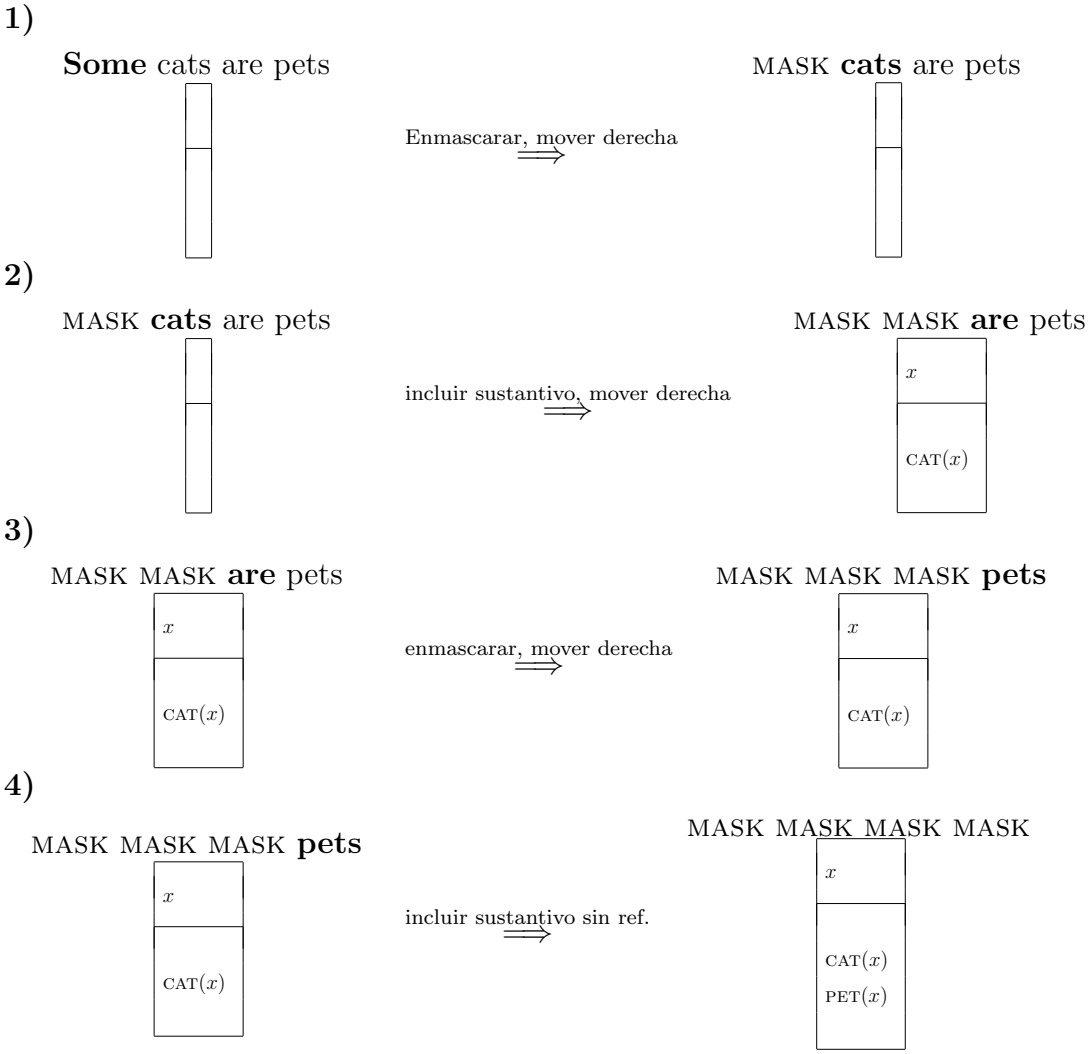

Figura 3: Ejemplo del procesamiento bajo las acciones (simplificadas) del entorno para la generación de la DRS para la frase "Some cats are pets". Fuente: Autoría propia.

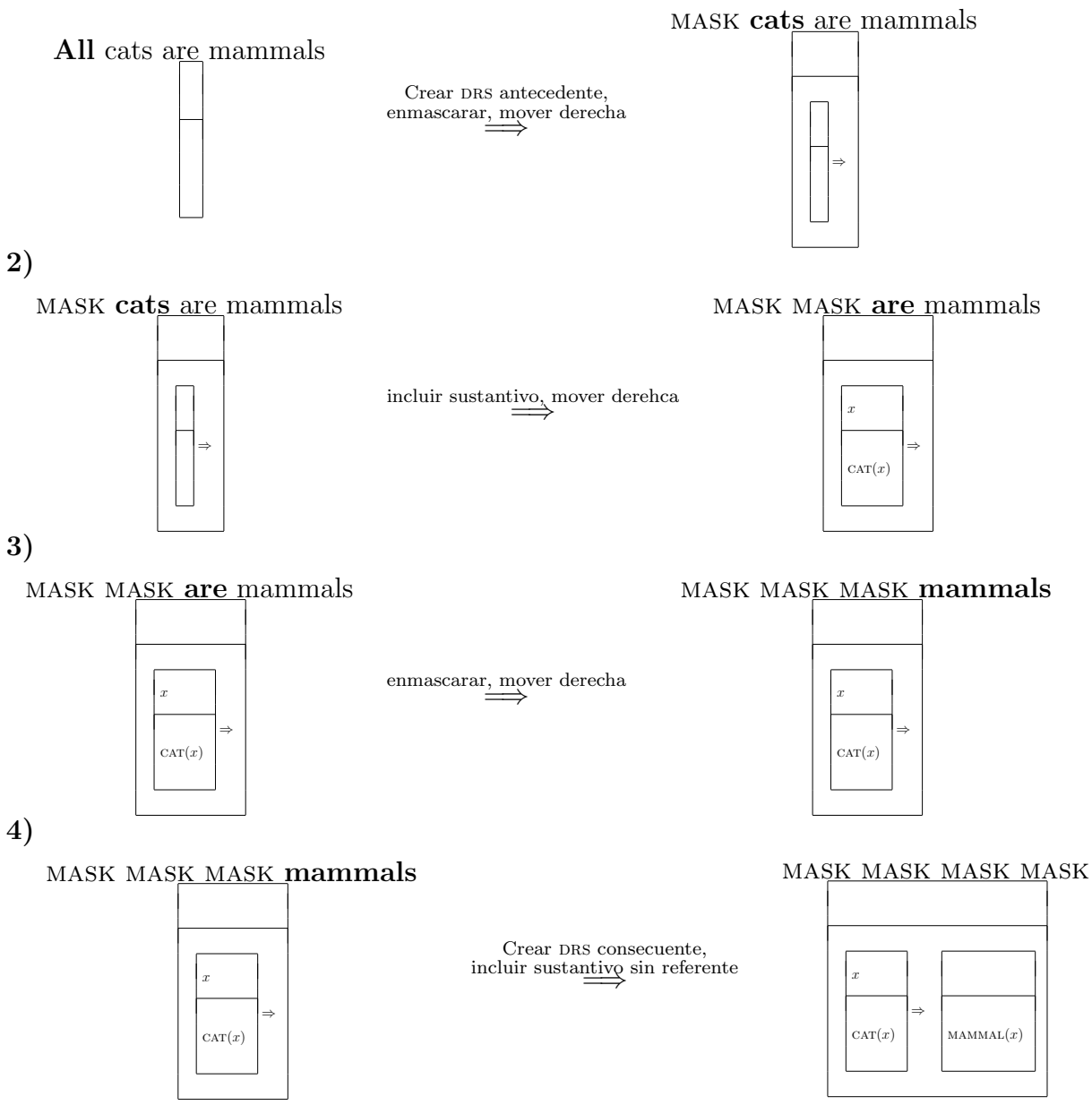

Figura 4: Ejemplo del procesamiento bajo las acciones (simplificadas) del entorno para la generación de la DRS para la frase "All cats are mammals". Fuente: Autoría propia.

- Si el índice en la secuencia del discurso se encuentra sobre una palabra ya procesada, es decir, marcada como MASK, el agente no puede realizar ninguna acción que procese la palabra sobre el índice actual.
- El agente no puede exceder el máximo nivel de profundidad si este fue definido, es decir, el agente no puede realizar ninguna acción de creación si su nivel actual de profundidad es el máximo.

Cualquier acción que no siga este conjunto de reglas, es considerada una acción incorrecta.

Teniendo en cuenta las reglas del entorno, la función de recompensa se define de acuerdo con las diferentes etapas de la simulación. Al finalizar la construcción de la DRS, el entorno verifica si la conclusión es una consecuencia lógica de la representación de las premisas. Si la respuesta es verdadera, el agente es recompensado con una puntuación de R_{final}; de lo contrario, incurre en una penalización de $-R_{\text{final}}$. Durante la fase de construcción de la DRS, el agente puede enfrentar diferentes resultados, que se describen a continuación:

1. Si la acción es incorrecta genera una penalización de $R_{a_{\text{incorrecta}}}$.
2. Si la acción no genera una modificación en la DRS y no es considerada una acción incorrecta no generan ningún costo, es decir, tienen un valor de 0.
3. Si la acción genera una modificación en la DRS y no es considerada una acción incorrecta, genera una recompensa F basada en una función de similitud respecto a la representación objetivo S_*, la representación de la iteración anterior S_{t-1} y la representación actual, S_t (ver más abajo).

En la figura 5, se puede observar de manera gráfica el proceso para la asignación del valor de recompensa.

Dada la complejidad inherente del entorno para la representación del lenguaje natural a una DRS, se planteo metodológicamente la definición de la función de recompensa como una adopción de la técnica de recompensa guiada (Reward Shaping) [44]. Esta técnica ayuda a generar una guía en el entrenamiento del agente hacia un comportamiento deseado de una manera más eficiente. De esta manera, el agente recibe recompensas o penalizaciones por acciones que son beneficiosas o perjudiciales a corto plazo, respectivamente. Para la definición de esta función, experimentalmente se garantizó la

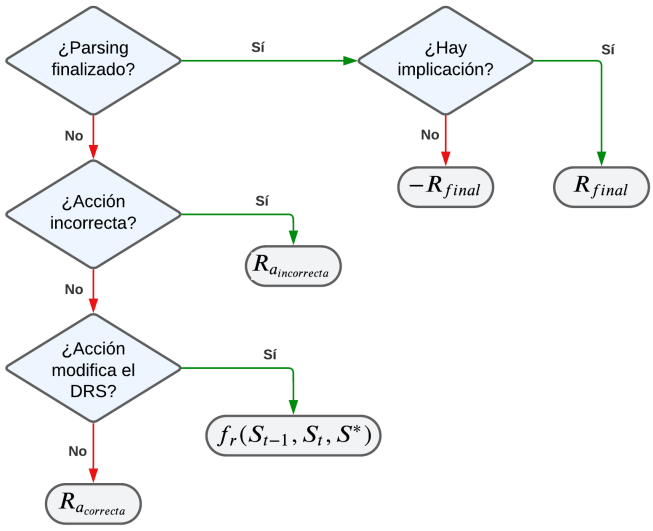

Figura 5: Diagrama de flujo de la función de recompensa. Fuente: Autoría propia.

convergencia de la política, es decir, se validó que la política óptima no se viera alterada por las recompensas guiadas. Para la implementación de este método, se debe definir una función potencial ϕ sobre el estado que asigne un valor indicando qué tan bueno o prometedor es [44, 45]. Dado el entorno, la función de potencia definida considera el estado objetivo y realiza una comparación que permite medir el avance porcentual entre dos estados consecutivos, como:

$$\phi(S_{t-1}, S_t, S^*) = 1 - \frac{d(S_t, S^*)}{d(S_{t-1}, S^*)}$$

Donde S_t es el estado actual, S_{t-1} es el estado anterior, S^* es el estado objetivo y $d(\cdot, \cdot)$ es la distancia de edición, también conocida como distancia Levenshtein [46]. Esta distancia calcula el costo mínimo de transformar una cadena de caracteres x a una cadena y. Dicho costo se estima como la cantidad de operaciones necesarias, las cuales incluyen eliminar, insertar o sustituir un carácter, para convertir la cadena x en la cadena y. Es importante destacar que la función de potencia mide el porcentaje de avance o retroceso en el proceso de representación, por lo cual, la función de recompensa estada dada por:

$$F(S_{t-1}, S_t, S^*) = \phi(S_{t-1}^{(iv)}, S_t^{(iv)}, S^{*(iv)}) + \alpha \cdot \phi(S_{t-1}^{(i)}, S_t^{(i)}, S^{*(i)})$$

Donde $S^{(iv)}$ el componente del estado crudo (no el *embedding*) que corresponde a la fórmula en FOL generada a partir del estado actual, y $S^{(i)}$ el componente del estado

crudo que corresponde al procesamiento actual de la frase. Por último, α denota un ponderador para el valor de la función de potencia del procesamiento de la frase. Esta tasa de descuento ayuda a regular la recompensa recibida por cada una de las acciones, de forma tal que el valor de recompensa o castigo sea mayormente influenciado por el estado de la representación de la formula FOL.

6.1.4. Implementación técnica del Entorno

El entorno ha sido desarrollado utilizando el lenguaje de programación *Python*. Se definió una clase principal denominada *ParserFol_1f*, encargada de representar el entorno personalizado. Esta clase gestiona todo el proceso relacionado al procesamiento de cada muestra para el entrenamiento, la definición del espacio de acciones y la aplicación de recompensas, todo en una interfaz compatible con el estándar Gymnasium, lo cual garantiza que el entorno pueda ser integrado fácilmente con diferentes algoritmos de RL.

Para el entrenamiento del agente se empleó el algoritmo DQN provisto por la librería *Stable-Baselines3*. Con el objetivo de mejorar la eficiencia del entrenamiento, se integró una adaptación del método Prioritized Experience Replay (PER).

6.2. Diseño y entrenamiento del agente

6.2.1. Conjunto de datos

FOLIO consta de 1430 ejemplos con conclusiones únicas y un conjunto de 487 premisas utilizadas para razonar deductivamente la validez de cada conclusión [43]. Aunque cada ejemplo del conjunto de datos está etiquetado y verificado por un motor de inferencia FOL, para el objetivo del presente trabajo se realizo un proceso de validación y filtrado de la representación en FOL tanto del conjunto de premisas como de las conclusiones.

En primer lugar fueron seleccionados los diez primeros ejemplos de menor longitud con base en las premisas, asegurando que la conclusión sea una implicación lógica, posteriormente, sobre estos 10 ejemplos, se realizó una validación de la consistencia en la representación lógica tanto de las premisas como de las conclusiones. A su vez, se verificó la veracidad de la implicación lógica entre las premisas y la conclusión bajo el motor de inferencia automatizado para lógica de primer orden implementado.

Una vez se ratificó la consistencia de este subconjunto de datos, se ordenó cada

ejemplo de menor a mayor longitud con base en las premisas y se adapto la estructura para poder utilizar cada ejemplo como un entorno para el aprendizaje del agente. Los ejemplo considerados se limitan a premisas y conclusiones que consideran unicamente conjunciones e implicaciones lógicas.

Un ejemplo de la estructura más sencilla utilizada para abordar el objeto del presente proyecto es la oración "*All cats are mammals*", donde la representación FOL definida por el conjunto de datos es: $\forall x\ (Cat(x) \rightarrow Mammal(x))$. Otro ejemplo de las estructuras utilizadas es la oración "*Some cats are pets*", donde la representación FOL dada es $\exists x(Cat(x) \wedge Pet(x))$.

6.2.2. Aprendizaje curricular

Dentro del aprendizaje automático de maquina, el aprendizaje curricular (CL, por sus siglas en inglés) es una estrategia de entrenamiento de orden significativo [47]. Esto quiere decir que el entrenamiento es llevado a cabo comenzando desde los casos más sencillos y gradualmente se presentan casos más complejos. Esta técnica puede ser aplicada en cualquiera de las principales etapas o componentes de cualquier modelo de aprendizaje automático, pues puede ser implementada sobre el conjunto de datos, el modelo, la tarea de interés o la métrica de rendimiento. Bajo este paradigma, diferentes autores han planteado distintos algoritmos o estrategias de CL que combinan la implementación de está técnica en diferentes etapas del proceso o definen una estructura y estrategia de implementación [48].

En el marco del aprendizaje por refuerzo, las categorías de CL que tienen mayor aplicación son Teacher-Student CL, en el cual el entrenamiento se divide en dos tareas, un modelo que aprende la tarea principia y un modelo auxiliar que evalúa y determina los parámetros del modelo principal; Self-Paced CL (SPCL), donde, con base en el aprendizaje y rendimiento del modelo, se ajusta dinámicamente el currículo, es decir, la complejidad del conjunto de entrenamiento; CL progresivo (PCL), donde gradualmente se incrementa la complejidad de las tareas presentadas al modelo; y por último, CL implícito, donde no se define explícitamente un currículum sino que este es consecuencia de una metodología especifica de entrenamiento [48–50].

Para el objeto del presente proyecto, se implementó una combinación de SPCL y PCL bajo la siguiente esquema iterativo:

1. **Criterio de dificultad:** medida utilizada para definir la complejidad de cada uno de los registros considerados en el conjunto de datos. Este criterio se usa para definir el orden de entrenamiento significativo. Dentro del marco del entorno de DRL implementado, el criterio de dificultad se define con base en la cantidad de acciones mínimas necesarias para lograr la representación DRS de una frase.

2. **Modelo de selección:** método que determina cual ejemplo debe ser considerado para el entrenamiento en el modelo actual. Debido a la complejidad inherente de la tarea del agente y el entrenamiento del mismo, esta selección se lleva a cabo manualmente bajo observaciones empíricas sobre los resultados de los modelos anteriores. Así mismo, en la iteración inicial, la selección de los parámetros de los modelos a entrenar se realiza priorizando la recompensa inmediata de las acciones y progresivamente se cambia este enfoque al largo plazo.

3. **Medición del rendimiento:** la medición del desempeño de cada uno de los modelos considerados en la etapa anterior se realiza analizando en primer lugar el aumento de la recompensa promedio por paso en el tiempo y, en segundo lugar, los valores Q y su relación con la recompensa que el modelo da a cada acción en el estado actual.

4. **Programador curricular:** en este proceso, se decide cuando se actualiza el currículo para utilizar el modelo o la iteración que ofrezca el mayor rendimiento global. Generalmente, el programador actúa de manera automática realizando una actualización en un intervalo de tiempo fijo (lineal o logarítmico). En nuestra metodología, este proceso no es automático. La decisión de actualización del currículo es manual y empírica, donde se tiene en cuenta la tarea actual dentro del sistema iterativo.

El PCL, permite aumentar gradualmente la complejidad de las tareas que se presentan al modelo, de esta forma, este tipo de CL es considerado en las dos primeras etapas del proceso iterativo. Por otro lado, el SPCL, busca ajustar el entrenamiento en función del rendimiento del agente y su progreso en el aprendizaje, lo que se ve reflejado en las dos ultimas etapas del proceso.

En la figura 6, se puede observar de manera gráfica el proceso de entrenamiento implementado haciendo uso de CL.

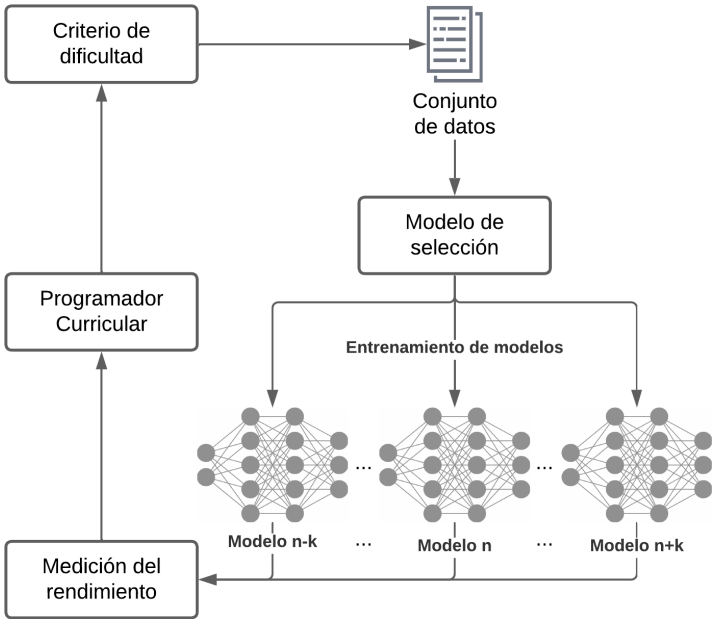

Figura 6: Estrategia utilizada para el aprendizaje curricular del modelo DRL, aplicada según el marco general del CL. Fuente: Autoría propia.

6.2.3. Proceso y técnicas de entrenamiento

El algoritmo de aprendizaje utilizado es DQN, con repetición de experiencias prioritarias (PER, por sus siglas en inglés). Los hiperparámetros del modelo se ajustaron siguiendo las recomendaciones del artículo de DQN [41]. En particular, el tamaño del búfer de experiencia y el intervalo de actualización de la red objetivo (*target network*) se definieron en función del número total de pasos de entrenamiento (*timesteps*), utilizando relaciones de 10:1 y 100:1, respectivamente. Es decir, si el entrenamiento se realiza durante $1,000$ iteraciones, el tamaño del búfer será de 100 y la red objetivo se actualizará cada 10 iteraciones. Estas proporciones permiten una sincronización adecuada entre exploración, explotación y estabilidad en el aprendizaje.

La incorporación de PER responde a la necesidad de mejorar la eficiencia del proceso de aprendizaje al evitar la selección uniforme de transiciones en el búfer de experiencias. Dada la complejidad del entorno, muchas de las transiciones almacenadas pueden ser poco informativas o redundantes, lo que ralentiza el proceso de convergencia del agente. Mediante la priorización basada en el error temporal-diferido (*TD error*), se asigna una mayor probabilidad de selección a las muestras que representan las experiencias que han generado una recompensa mayor significativa para el agente. Esta asignación permite

un ajuste más rápido y dirigido de la política [51]. Adicionalmente, esta estrategia contribuye a una representación más robusta del entorno.

Dada la dificultad de la tarea del entorno, el proceso de entrenamiento se llevó a cabo restringiendo el espacio de acciones a seis posibilidades: (i) mover a la derecha, (ii) enmascarar, (iii) incluir sustantivo, (iv) incluir sustantivo sin referente, (v) crear DRS antecedente, y (vi) crear DRS consecuente. La acción enmascarar y las acciones de tipo modificación incluyen de forma implícita la acción mover a la derecha. Además, si se ha ejecutado previamente una acción de creación de DRS, estas acciones inducen un ascenso de nivel en la profundidad de representación.

Esta restricción del espacio de acciones fue fundamental para estabilizar el entrenamiento en etapas tempranas y facilitar una exploración más eficiente y dirigida del entorno. Con base en la estrategia de entrenamiento descrita en la sección anterior, se implementó un entrenamiento progresivo de la dificultad de las tareas de aprendizaje del agente bajo las siguientes etapas:

1. Se inició con la oración estructuralmente más sencilla, enfocando la tarea del agente en la aproximación de los valores Q a un solo paso. El objetivo de esta primera etapa es que el agente aprenda a seleccionar la acción con mayor recompensa inmediata, sin considerar consecuencias a largo plazo.

2. Posteriormente, conservando la misma oración y validando que los resultados del paso anterior correspondan al mismo orden de los valores definidos por la función de recompensa, se incrementó la dificultad de la tarea cambiando el máximo de turnos a dos pasos. Esta etapa busca permitir que el agente aprenda la primera secuencia de acciones que conduce a la mayor recompensa en el corto plazo, comenzando a desarrollar el primer entendimiento de acciones secuenciales.

3. Con el conocimiento adquirido en las dos etapas previas, se aumentó el numero máximo de turnos progresivamente hasta que el agente sea capaz de generar una representación lógica adecuada para la estructura oracional dada. De igual manera, conforme el número máximo de turnos aumenta, el enfoque temporal del agente se ajusta a ello. Esta etapa puede entenderse como una inicialización informada de los pesos de la red, aprovechando el conocimiento adquirido previamente para facilitar la convergencia hacia políticas más útiles.

4. Posteriormente, se integró la segunda oración estructuralmente más sencilla, se entrenó al agente siguiendo los pasos 1, 2 y 3. En las diferentes iteraciones de entrenamiento, se mantuvo una probabilidad dada de escoger la segunda frase respecto a la primera. Esta etapa introduce complejidad semántica adicional mientras se preserva la familiaridad estructural de la primera oración, permitiendo una transición controlada en la dificultad del entorno.

5. Finalmente, una vez completadas estas fases iniciales —diseñadas para guiar al agente en la adquisición de una política base robusta— se entrenó al agente con oraciones estructuralmente iguales. Estas incluyen construcciones conjuntivas e implicatorias, con el objetivo de fomentar la generalización en la generación de representaciones en FOL. Esta fase se desarrolló de forma regular a lo largo del entrenamiento, promoviendo la transferencia y consolidación de lo aprendido en estructuras semánticas más diversas.

6.2.4. Deep Q-Network

El objetivo de la red, como se ha mencionado previamente, es que el agente pueda decidir qué acción tomar dado el estado actual. Para ello, la entrada de la red es un tensor de dimensiones 1538×1, obtenido a partir del proceso de codificación definido en la Sección 6.1.1. La arquitectura de la red consta de dos capas lineales: la primera de tamaño 1024 y la segunda de tamaño 64. Entre cada capa lineal se aplica una función de activación ReLU. De esta forma, los embeddings de 1538 dimensiones son proyectados primero a 1024 y luego a 64 dimensiones. Finalmente, la salida de la red se adapta al número de acciones disponibles en el entorno, resultando en una salida de 6 dimensiones, donde cada dimensión representa una de las posibles acciones que el agente puede tomar.

7 RESULTADOS Y DISCUSIÓN

7.1. Protocolo de entrenamiento

Siguiendo el proceso descrito en la sección 6.2.3 se obtuvieron los siguientes resultados. La primera frase de entrenamiento fue "Some cats are pets". Esta frase es estructuralmente la más sencilla debido a que requiere unicamente de 4 acciones para llegar a su representación. Para llevar a cabo el proceso de entrenamiento, se definió la tarea del agente en cada iteración bajo el siguiente esquema de entrenamiento:

Entrenamiento	**Turnos máximos**	**Factor de descuento**	**Pasos de tiempo**
1	2	0.1	6 000
2	4	0.3	25 000

Tabla 1: Parámetros del esquema de entrenamiento para la frase "Some cats are pets"

Obsérvese que el primer entrenamiento utiliza un factor de descuento $\gamma = 0{,}1$ para priorizar las recompensas a corto plazo, en concordancia con el número máximo de turnos. La elección de dos turnos se debe a que, en este caso, la acción que genera la mayor recompensa inmediata en un solo paso no coincide con la primera acción de la secuencia de representación esperada.

El segundo entrenamiento tiene como objetivo que el agente aprenda la secuencia completa de acciones necesarias para una representación adecuada de la frase. En este caso, la selección de un factor de descuento bajo (menor a 0.5) se debe a que, si se utiliza un valor que prioriza excesivamente la recompensa a largo plazo, el agente tiende a aprender una representación subóptima durante las primeras iteraciones. Es decir, el agente aprende una política que lo conduce a un máximo local. En la figura 7 se puede observar los resultados del entrenamiento.

Es importante exaltar que, el primer entrenamiento utiliza un factor de descuento $\gamma = 0{,}1$ para priorizar las recompensas en el corto plazo en concordancia con el número máximo de turnos. La selección de dos turnos es debido a que, para este caso, la acción que genera la mayor recompensa inmediata en un paso, no concuerda con la primer acción de la secuencia de representación esperada. El segundo entrenamiento busca que el agente aprenda la secuencia de acciones necesarias para la representación adecuada de la frase. En este caso, la selección de un factor de descuento bajo (menor al 0.5) se

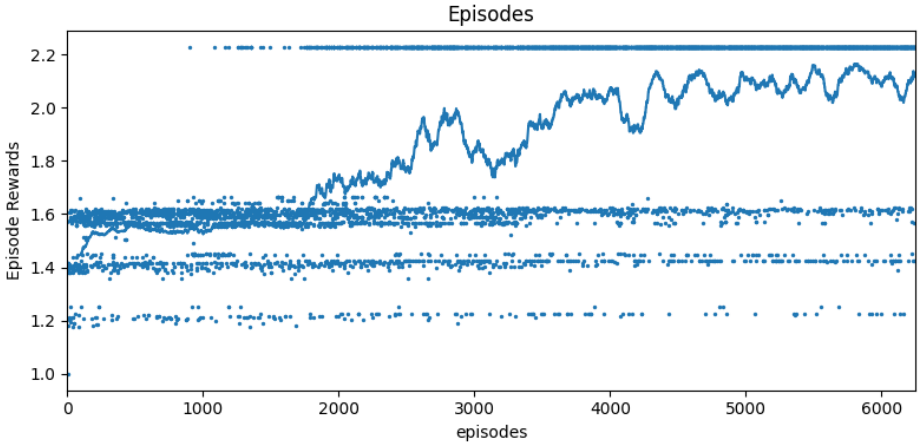

Figura 7: Resultados del esquema de entrenamiento definido en la tabla 1. Gráfica de la recompensa recibida por episodio. Fuente: Autoría propia.

debe a que, si se usa un factor de descuento que prioriza la recompensa a largo plazo, en las primeras iteraciones del modelo el agente aprende una representación subóptima. Esto quiere decir, que el agente aprende una política que lo lleva a un máximo local. Como se puede observar en la gráfica 7, El entrenamiento, teniendo en cuenta que se esta usando una politice ϵ-*greedy* muestra una mejora consistente conforme el avance de episodios, logrando así, capturar el comportamiento esperado.

Posterior a este entrenamiento, se sumo la frase "All cats are mammals". Esta frase requiere de 7 acciones para llegar a la representación esperada. Para ello, se siguió el siguiente esquema de entrenamiento:

Entrenamiento	**Turnos máximos**	**Factor de descuento**	**Pasos de tiempo**
3	1	0.0	10 000
4	3	0.3	20 000
5	5	0.3	500 000
6	7	0.4	250 000

Tabla 2: Parámetros del esquema de Entrenamiento para las frases "All cats are mammals" y "Some cats are pets"

Este proceso de entrenamiento es seguido de la primera etapa descrita en la tabla 1. Dado que el entorno cuenta con dos frases a representar, la selección aleatoria de la frase para cada simulación se modifico para que, con un 60 % de probabilidad, fuera

seleccionada la frase “All cats are mammals”. El objetivo de esta modificación es priorizar el entrenamiento del agente sobre el nuevo entorno incorporado, pero, previniendo a su vez, que se presente un olvido catastrófico (Catastrophic Forgetting) en el agente.

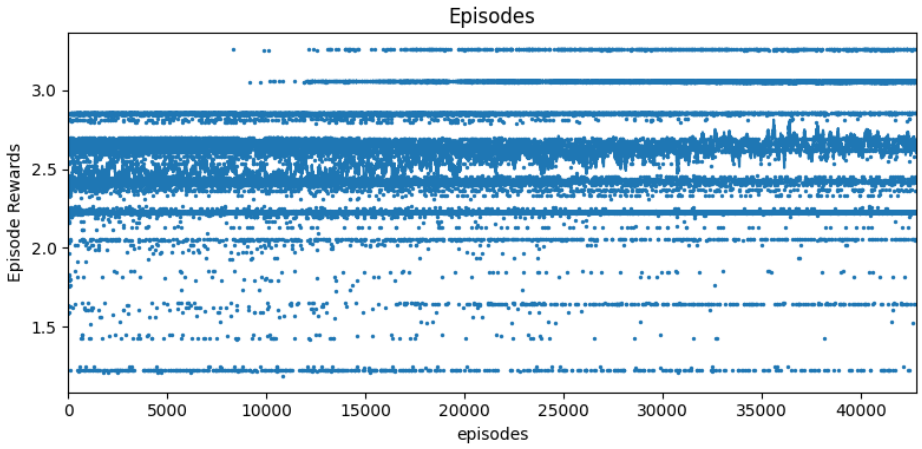

Figura 8: Resultados del esquema de entrenamiento definido en la tabla 2. Gráfica de la recompensa recibida por episodio. Fuente: Autoría propia.

Note que en la primera iteración de entrenamiento, se utilizó un máximo de turnos de 1 y un factor de descuento $\gamma = 0$. La tarea para esta etapa, es que el agente aprenda la acción que le retorna una mayor recompensa en el inmediato plazo con el fin de que aprenda a diferenciar entre las dos estructuras de las oraciones presentadas.

En la gráfica 8, se puede observar como a medida que los episodios de entrenamiento aumentan la recompensa tienen a la recompensa esperada de las dos estructuras deseadas, lo que le permite al agente generar la secuencia de acciones correspondiente para cada frase.

Para evaluar la escalabilidad del modelo, se incorporaron de a una frase para cada caso, es decir, del estilo “Some x are y” y “All x are y”. El proceso se realizó hasta incorporar 4 frases de cada tipo estructural. El esquema implementado de entrenamiento se describe a continuación:

Note que este esquema de entrenamiento conserva las características principales de los esquemas presentados en las Tablas 1 y 2. En el primer entrenamiento, se utiliza un factor de descuento $\gamma = 0$ para que el agente aprenda la acción que le otorga la mayor recompensa a corto plazo. Posteriormente, la tarea del agente se centra en aprender la secuencia de acciones de tres pasos que maximiza su recompensa. El último entrenamiento, permite el aprendizaje de la secuencia completa de acciones necesarias

Entrenamiento	Turnos máximos	Factor de descuento	Pasos de tiempo
0	1	0	5 000
1	3	0.3	50 000
2	7	0.4	N

Tabla 3: Parámetros del esquema de Entrenamiento para la inclusión de nuevos ejemplos.

para representar cada una de las frases consideradas.

Es importante destacar que la cantidad de pasos de entrenamiento N de la tercera etapa está acotada por intervalos de $200,000$ con una cantidad mínima de pasos correspondiente a $200,000$. De esta forma, los intervalos avanzan proporcionalmente al número de oraciones bajo la siguiente fórmula:

$$200,000 \cdot (n-1) \leq N \leq 200,000 \cdot n$$

Donde n representa el número de frases consideradas en el entorno. En los modelos desarrollados, cuando el entorno cuenta con 2 frases de representación (es decir, 4 ejemplos: 2 del tipo "Some" y 2 del tipo "All"), el agente se entrenó durante $350,000$ pasos. Para el caso de 3 frases (3 de cada tipo), se entrenó durante $500,000$ pasos. Con 4 frases, el entrenamiento fue de $600,000$ pasos, y para 5 frases, el agente fue entrenado durante $850,000$ pasos. El conjunto de datos utilizado en esta etapas de entrenamiento es dado por las siguientes frases:

1. Some cats are pets. All cats are mammals.
2. Some cars are fast. All cars are vehicles.
3. Some flowers are red. All flowers are plants.
4. Some phones are smart. All phones are gadgets.
5. Some shoes are comfortable. All shoes are footwear.

La primera frase corresponde a uno de los ejemplos más sencillos del conjunto de datos FOLIO que enmarca la representación de implicaciones y conjunciones. Los demás ejemplos, son adaptaciones del ejemplo de FOLIO creadas únicamente con el fin de entrenar y evaluar el modelo para la creación de este tipo de representaciones.

7.2. Capacidades de generalización

7.2.1. Análisis cuantitativo

Examinamos de manera cuantitativa las capacidades de generalización del modelo, con base en la cantidad de ejemplos sobre los cuales se entrenó. Para este fin, se estableció un conjunto de prueba definido por los siguientes ejemplos:

1. Some computers are slow. All computers are machines.
2. Some watches are luxury. All watches are accessories.
3. Some paintings are famous. All paintings are artworks.
4. Some clothes are fashionable. All clothes are apparel.
5. Some jewelry are expensive. All jewelry are ornaments.
6. Some buildings are tall. All buildings are structures.

Se validó la capacidad de cada modelo entrenado para generar la DRS correspondiente para cada frase de prueba. Los resultados se encuentran consolidados en la tabla 4. Se puede observar que el modelo entrenado con 5 frases es capaz de desempeñarse de manera satisfactoria sobre una frase de cada tipo estructural. Este resultado respalda la validez del esquema de entrenamiento desarrollado y da un buen indicio para su capacidad de generalización.

Entrenamiento	1	2	3	4	5
Some	0 %	0 %	0 %	16, 6 %	16, 6 %
All	0 %	0 %	0 %	0 %	16, 6 %

Tabla 4: Porcentaje de éxito de cada modelo en el conjunto de prueba. El entrenamiento corresponde con el número de frases consideradas dentro del modelo bajo la metodología defina en las tablas 1 y 2 en los casos 1 y 2 y bajo la metodología descrita en la tabla 3 para los demás casos. El porcentaje de 16, 6 % indica que el modelo fue capaz de representar exitosamente 1 de las 6 frases consideradas

Los resultados del entrenamiento del modelo con mayor capacidad de generalización se presentan en la gráfica 9, donde se muestra la recompensa promedio obtenida en una ventana de 1, 000 pasos. Se observa que el comportamiento de la curva de aprendizaje es

crecientemente positivo a lo largo de todo el entrenamiento, lo cual sugiere que extender el entrenamiento durante más pasos puede mejorar la capacidad de generalización del agente. No obstante, entrenar el modelo durante una mayor cantidad de pasos implica un alto consumo de recursos computacionales.

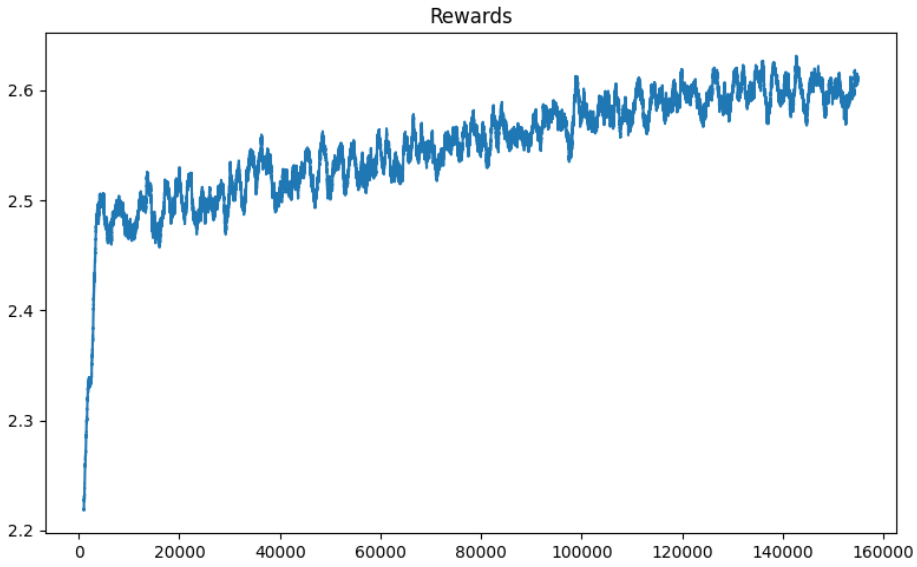

Figura 9: Recompensa promedio por ventana de 1 000 episodio para el modelo con 5 frases, entrenado según el esquema de la tabla 4. Fuente: Autoría propia.

7.2.2. Análisis cualitativo

Con el propósito de evaluar la capacidad de generalización del modelo desarrollado, se llevó a cabo un análisis desde la perspectiva de la representación vectorial de los estados generados por el agente. Para ello, se seleccionaron tres frases representativas definidas en el entorno:

- Some cats are pets. All cats are mammals.
- Some flowers are red. All flowers are plants.
- Some paitings are famous. All paintings are artworks.

Para cada una de estas frases, y considerando el conjunto de acciones requeridas para su representación, se almacenaron los estados correspondientes y se proyectaron sus embeddings. Para ello, se utilizó la técnica de reducción de dimensiones t-SNE.En la figura 10 se presenta el conjunto de embeddings obtenidos para las frases de tipo “All x

are y". La gráfica evidencia una separación clara entre los grupos de estados asociados a cada frase, lo cual favorece a que el modelo sea capaz de identificar y mantener estructuras diferenciadas. Asimismo, se observa una consistencia en la organización interna de las secuencias de acciones, lo que ayuda a que el agente sea capaz de aprender representaciones estructurales coherentes y reutilizables entre ejemplos con patrones lógicos similares bajo.

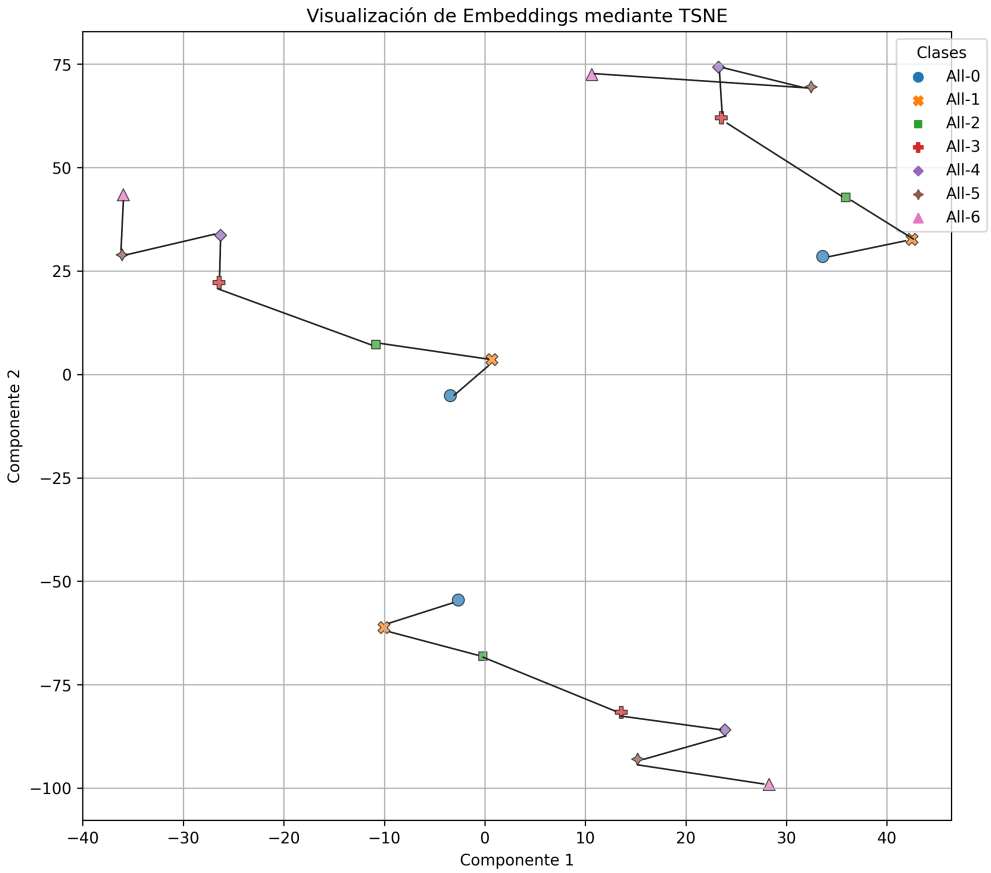

Figura 10: Representación en 2 dimensiones de los embeddings que genera cada estado dada la secuencia de acciones necesarias para la representación de frases de tipo "All x are y". Fuente: Autoría propia.

En la figura 11 se presenta una visualización para las frases del tipo "Some x are y", lo que implica una secuencia de acciones para la generación de la DRS más corta. Este caso permite apreciar con mayor claridad la organización interna de los estados generados. De esta forma, sin importar la complejidad (numero de acciones necesarias), se conserva la separación entre los distintos ejemplos, lo que refuerza la hipótesis de que el modelo de representación de estados, favorece al agente parar lograr abstraer la estructura lógica

de estas frases de forma robusta. La capacidad del modelo de embeddings para generar esta distinción, incluso en representaciones más simples, sugiere un aprendizaje sólido en el modelo DRL de los patrones involucrados.

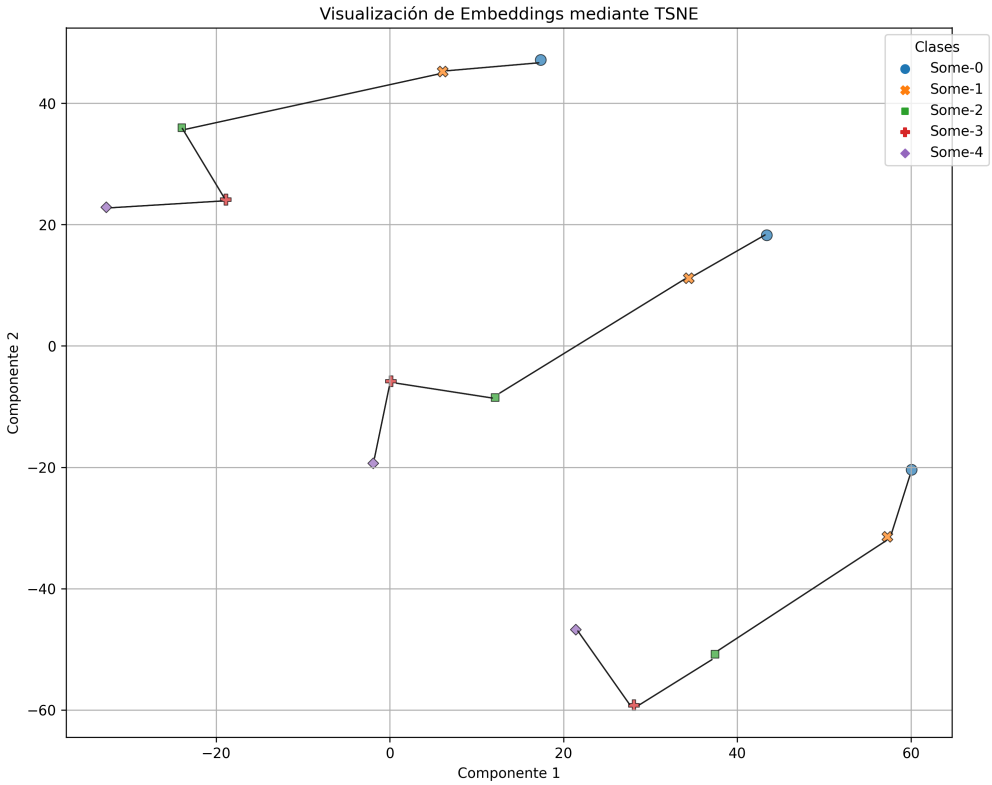

Figura 11: Representación en 2 dimensiones de los embeddings que genera cada estado dada la secuencia de acciones necesarias para la representación de frases de tipo “Some x are y”. Fuente: Autoría propia.

Por último, la figura 12 muestra la representación individual de cada acción dentro de las secuencias correspondientes a cada frase. En este caso, se evidencia una separación aún más clara entre las distintas estructuras, con agrupamientos bien definidos que reflejan tanto la identidad de la frase como la función de cada acción en el proceso de construcción de la DRS. Este patrón refuerza la conclusión de que el modelo de embeddings asiste y apoya la capacidad del modelo para hacer una distinción correcta entre las frases con estructuras similares y asignar una representación distintiva a cada paso del proceso requerido para su representación.

En conjunto, estos resultados respaldan la capacidad del agente para generalizar en tareas de representación lógica. La coherencia observada entre los distintos niveles de análisis sugiere que el modelo de representación de estados aporta a la capacidad de

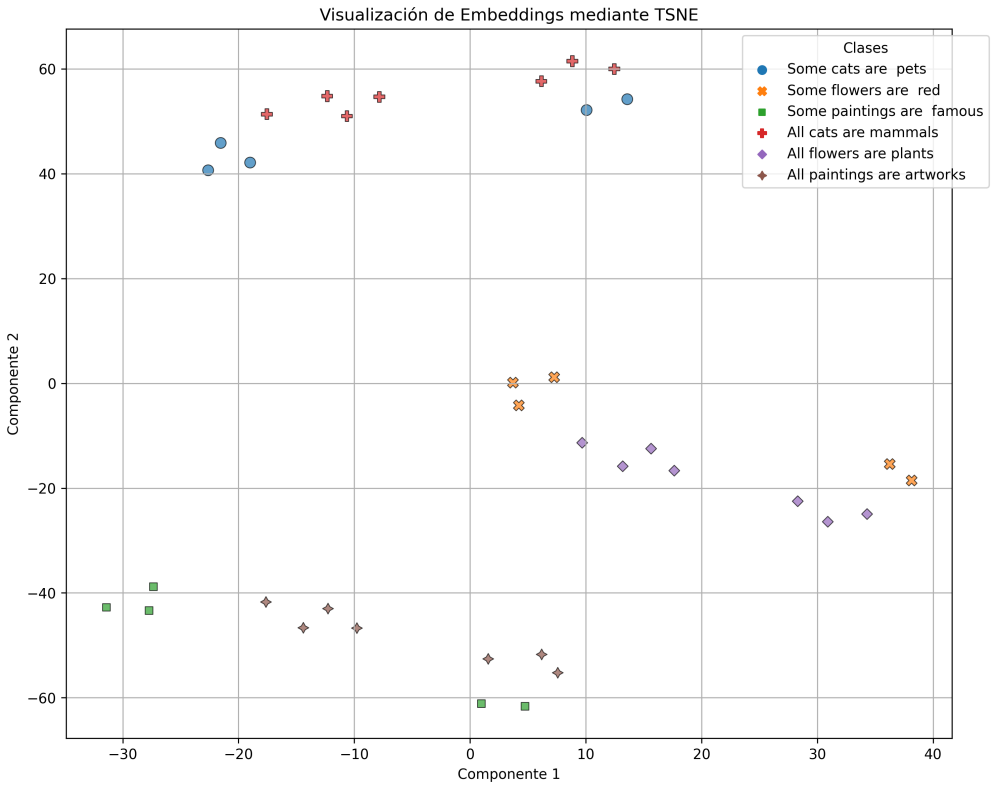

Figura 12: Embeddings some y all. Fuente: Autoría propia

aprender un espacio de representación significativo por parte del modelo DRL. Siendo capaz de capturar las relaciones entre elementos lógicos y de reutilizar patrones aprendidos en nuevos contextos.

7.3. Discusión

El protocolo desarrollado ha demostrado ser efectivo en la generación de representaciones lógicas mediante aprendizaje por refuerzo profundo. No obstante, el proceso de entrenamiento resulta ser altamente oneroso desde el punto de vista computacional. Este fenómeno se presenta debido a que la cantidad de pasos necesarios para entrenar el modelo crece de manera exponencial conforme aumenta la complejidad en la representación y distinción de las estructuras lingüísticas. Esta observación sugiere la necesidad de explorar más estrategias de optimización del entorno, como un barrido de hiperparámetros para encontrar los valores óptimos y la automatización de la estrategia de aprendizaje curricular.

En cuanto a las capacidades de generalización, los resultados preliminares indican un comportamiento prometedor. Las propiedades geométricas observadas en los espacios

de embeddings generados parecen facilitar la agrupación y separación de conceptos relacionados, lo cual favorece la transferencia del conocimiento aprendido a nuevas estructuras. Este comportamiento abre una línea de análisis para las demás estructuras lógicas contenidas en el conjunto de datos de FOLIO.

Como trabajo futuro, se propone adelantar un estuido más amplio y completo sobre las capacidades de generalización del agente. También, se plantea extender el entorno incorporando otras estructuras lógicas presentes en FOLIO, tales como predicados con constantes (ej. "Predicado(constante)"), el uso de negaciones, y formas plurales. Esta ampliación permitiría evaluar la robustez del agente frente a una mayor variedad lógica y gramatical, y sentar las bases para una aplicación más general en entornos naturales.

8 CONCLUSIONES

En este trabajo se desarrolló un parser semántico basado en aprendizaje por refuerzo profundo, con el propósito de representar oraciones en lenguaje natural como fórmulas de la lógica de primer orden, y con ello, permitir la evaluación de deducciones sobre las representaciones. Esta propuesta se plantea como una extensión del trabajo de [1], con mejoras en la arquitectura del entorno de entrenamiento, la formulación del sistema de recompensas y la cobertura semántica del lenguaje.

Para ello, se extendieron el conjunto de reglas de representacion para poder representar silogismos con formas conjuntivas e implicatorias. Además, se reformuló el sistema de recompensas para reflejar la validez lógica de las representaciones generadas y favorecer el entrenamiento del agente. Por otro lado, se definió un esquema de aprendizaje con el fin de manejar la complejidad inherente en el proceso de aprendizaje del agente.

Los resultados que se obtuvieron son prometedores. El análisis del desempeño del agente y la representación vectorial de los estados demuestra que el modelo tiene el potencial de generalizar más allá de los ejemplos entrenados, lo cual valida la viabilidad del enfoque propuesto y sugiere que puede ser extendido a representaciones más complejas bajo el estudio robusto de la función de recompensas, los esquemas de aprendizaje propuestos y la representación vectorial de los estados.

En trabajos futuros, deseamos extender las capacidades del parser para abarcar un mayor número de estructuras lógicas. También, esperamos explorar técnicas para la reducción del costo computacional; implementar un estudio sistemático sobre las capacidades de generalización; e incorporar la representación simbólica del agente con motores de deducción formal para generar conclusiones automáticas.

9 REFERENCIAS

[1] J. Piza-Londono and E. Andrade-Lotero, "Leveraging semantic parsing using text embeddings and reinforcement learning," in *2024 IEEE Latin American Conference on Computational Intelligence, LA-CCI 2024 - Proceedings*, A. Orjuela-Canon, Ed. United States: Institute of Electrical and Electronics Engineers Inc., 2024, publisher Copyright: © 2024 IEEE.; 2024 IEEE Latin American Conference on Computational Intelligence, LA-CCI 2024 ; Conference date: 13-11-2024 Through 15-11-2024.

[2] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2023. [Online]. Available: https://arxiv.org/abs/1706.03762

[3] D. Jurafsky and J. Martin, *Speech and Language Processing*. Prentice Hall, 2008.

[4] J. Eisenstein, *Introduction to Natural Language Processing*. MIT Press, 2019.

[5] S. Ozdemir, *Quick Start Guide to Large Language Models: Strategies and Best Practices for ChatGPT, Embeddings, Fine-Tuning, and Multimodal AI*, 2nd ed. Addison-Wesley Professional, October 2024.

[6] L. Huang, W. Yu, W. Ma, W. Zhong, Z. Feng, H. Wang, Q. Chen, W. Peng, X. Feng, B. Qin, and T. Liu, "A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions," *ACM Transactions on Information Systems*, vol. 43, no. 2, p. 1–55, Jan. 2025. [Online]. Available: http://dx.doi.org/10.1145/3703155

[7] Y. Chang, X. Wang, J. Wang, Y. Wu, L. Yang, K. Zhu, H. Chen, X. Yi, C. Wang, Y. Wang, W. Ye, Y. Zhang, Y. Chang, P. S. Yu, Q. Yang, and X. Xie, "A survey on evaluation of large language models," *ACM Trans. Intell. Syst. Technol.*, vol. 15, no. 3, Mar. 2024. [Online]. Available: https://doi.org/10.1145/3641289

[8] F. Xu, Q. Lin, J. Han, T. Zhao, J. Liu, and E. Cambria, "Are large language models really good logical reasoners? a comprehensive evaluation and beyond," 2024. [Online]. Available: https://arxiv.org/abs/2306.09841

[9] H. Kamp and U. Reyle, *From Discourse to Logic: Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory*. Dordrecht, The Netherlands: Springer, 1993.

[10] L. Gamut, *Lógica, lenguaje y significado: lógica intensional y gramática lógica*, E. Andrade-Lotero and C. M. (translators), Eds. Bogotá: Editorial Universidad del Rosario, 2010.

[11] D. McDermott, "A critique of pure reason," *Computational Intelligence*, vol. 3, pp. 151–160, 1987.

[12] M. Minsky, "A framework for representing knowledge," in *The Psychology of Computer Vision*, P. Winston, Ed. McGraw-Hill, 1975.

[13] D. Urrutia. Qué es un analizador sintáctico o parser — definición, significado y para qué sirve. [Online]. Available: https://www.arimetrics.com/glosario-digital/analizador-sintactico-parser

[14] Y. Li and A. Risteski. The limitations of limited context for constituency parsing. [Online]. Available: https://arxiv.org/abs/2106.01580

[15] IBM. Large language models. [Online]. Available: https://www.ibm.com/topics/large-language-models

[16] I. Mirzadeh, K. Alizadeh, H. Shahrokhi, O. Tuzel, S. Bengio, and M. Farajtabar, "Gsm-symbolic: Understanding the limitations of mathematical reasoning in large language models," 2024. [Online]. Available: https://arxiv.org/abs/2410.05229

[17] Z. Wu, L. Qiu, A. Ross, E. Akyürek, B. Chen, B. Wang, N. Kim, J. Andreas, and Y. Kim. Reasoning or reciting? exploring the capabilities and limitations of language models through counterfactual tasks. [Online]. Available: https://arxiv.org/abs/2307.02477

[18] V. Basmov, Y. Goldberg, and R. Tsarfaty. Simple linguistic inferences of large language models (llms): Blind spots and blinds. [Online]. Available: https://arxiv.org/abs/2305.14785

[19] D. Gabbay and F. Guenthner, Eds., *Handbook of Philosophical Logic*. Springer Finance, 2007, 14 volumes, from 2001 to 2007.

[20] A. Kamath and R. Das, "A survey on semantic parsing," in *Automated Knowledge Base Construction (AKBC)*, 2019. [Online]. Available: https://openreview.net/forum?id=HylaEWcTT7

[21] D. Bekki, "Combinatory categorial grammar as a substructural logic," in *New Frontiers in Artificial Intelligence*, T. Onada, D. Bekki, and E. McCready, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 16–29.

[22] C. Liang, J. Berant, Q. Le, K. D. Forbus, and N. Lao, "Neural symbolic machines: Learning semantic parsers on freebase with weak supervision," 2017. [Online]. Available: https://arxiv.org/abs/1611.00020

[23] X. Lu, J. Liu, Z. Gu, H. Tong, C. Xie, J. Huang, Y. Xiao, and W. Wang, "Parsing natural language into propositional and first-order logic with dual reinforcement learning," in *Proceedings of the 29th International Conference on Computational Linguistics*, N. Calzolari, C.-R. Huang, H. Kim, J. Pustejovsky, L. Wanner, K.-S. Choi, P.-M. Ryu, H.-H. Chen, L. Donatelli, H. Ji, S. Kurohashi, P. Paggio, N. Xue, S. Kim, Y. Hahm, Z. He, T. K. Lee, E. Santus, F. Bond, and S.-H. Na, Eds. Gyeongju, Republic of Korea: International Committee on Computational Linguistics, Oct. 2022, pp. 5419–5431. [Online]. Available: https://aclanthology.org/2022.coling-1.481/

[24] Z. Zheng, Y. Wang, Y. Huang, S. Song, M. Yang, B. Tang, F. Xiong, and Z. Li, "Attention heads of large language models: A survey," 2024. [Online]. Available: https://arxiv.org/abs/2409.03752

[25] Y. Zhou, J. Ye, Z. Ling, Y. Han, Y. Huang, H. Zhuang, Z. Liang, K. Guo, T. Guo, X. Wang, and X. Zhang, "Dissecting logical reasoning in llms: A fine-grained evaluation and supervision study," 2025. [Online]. Available: https://arxiv.org/abs/2506.04810

[26] L. Pan, A. Albalak, X. Wang, and W. Y. Wang, "Logic-lm: Empowering large language models with symbolic solvers for faithful logical reasoning," 2023. [Online]. Available: https://arxiv.org/abs/2305.12295

[27] S. CS224N, C. Project, H. Elezabi, and B. Akoush, "Prototype-then-refine: A neurosymbolic approach for improved logical reasoning with llms," in *Stanford*

CS224N, 2025. [Online]. Available: https://api.semanticscholar.org/CorpusID:269325536

[28] P. Shojaee*†, I. Mirzadeh*, K. Alizadeh, M. Horton, S. Bengio, and M. Farajtabar, "The illusion of thinking: Understanding the strengths and limitations of reasoning models via the lens of problem complexity," 2025. [Online]. Available: https://ml-site.cdn-apple.com/papers/the-illusion-of-thinking.pdf

[29] S. Yao, D. Yu, J. Zhao, I. Shafran, T. L. Griffiths, Y. Cao, and K. Narasimhan, "Tree of thoughts: Deliberate problem solving with large language models," 2023. [Online]. Available: https://arxiv.org/abs/2305.10601

[30] T. He, H. Li, J. Chen, R. Liu, Y. Cao, L. Liao, Z. Zheng, Z. Chu, J. Liang, and M. Liu, "A survey on complex reasoning of large language models through the lens of self-evolution," 02 2025.

[31] J. Lee and J. Hockenmaier, "Evaluating step-by-step reasoning traces: A survey," 2025. [Online]. Available: https://arxiv.org/abs/2502.12289

[32] Q. Chen, L. Qin, J. Liu, D. Peng, J. Guan, P. Wang, M. Hu, Y. Zhou, T. Gao, and W. Che, "Towards reasoning era: A survey of long chain-of-thought for reasoning large language models," 2025. [Online]. Available: https://arxiv.org/abs/2503.09567

[33] R. S. Sutton and A. G. Barto, *Reinforcement Learning, second edition: An Introduction*. Cambridge, MA: MIT Press, 2018.

[34] R. Paulus, C. Xiong, and R. Socher, "A deep reinforced model for abstractive summarization," 2017. [Online]. Available: https://arxiv.org/abs/1705.04304

[35] A. Grissom II, H. He, J. Boyd-Graber, J. Morgan, and H. Daumé III, "Don't until the final verb wait: Reinforcement learning for simultaneous machine translation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, A. Moschitti, B. Pang, and W. Daelemans, Eds. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1342–1352. [Online]. Available: https://aclanthology.org/D14-1140/

[36] D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke, and S. Levine, "Qt-opt: Scalable deep

reinforcement learning for vision-based robotic manipulation," 2018. [Online]. Available: https://arxiv.org/abs/1806.10293

[37] E. Choi, D. Hewlett, A. Lacoste, I. Polosukhin, J. Uszkoreit, and J. Berant, "Hierarchical question answering for long documents," 2017. [Online]. Available: https://arxiv.org/abs/1611.01839

[38] V. Uc-Cetina, N. Navarro-Guerrero, A. Martin-Gonzalez, C. Weber, and S. Wermter, "Survey on reinforcement learning for language processing," *Artificial Intelligence Review*, vol. 56, no. 2, pp. 1543–1575, Jun. 2022.

[39] B. Jang, M. Kim, G. Harerimana, and J. W. Kim, "Q-learning algorithms: A comprehensive classification and applications," *IEEE Access*, vol. 7, pp. 133 653–133 667, 2019.

[40] V. François-Lavet, P. Henderson, R. Islam, M. G. Bellemare, and J. Pineau, "An introduction to deep reinforcement learning," *Foundations and Trends® in Machine Learning*, vol. 11, no. 3-4, pp. 219–354, Jan. 2018.

[41] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," 2013. [Online]. Available: https://arxiv.org/abs/1312.5602

[42] Y. Keneshloo, T. Shi, N. Ramakrishnan, and C. K. Reddy, "Deep reinforcement learning for sequence to sequence models," 2019. [Online]. Available: https://arxiv.org/abs/1805.09461

[43] S. Han, H. Schoelkopf, Y. Zhao, Z. Qi, M. R. W. Zhou, J. Coady, D. Peng, Y. Qiao, L. Benson, L. Sun, A. W.-S. H. Szabo, E. Zubova, M. Burtell, J. Fan, Y. Liu, B. Wong, M. Sailor, A. Ni, L. Nan, J. Kasai, T. Yu, R. Zhang, A. R. Fabbri, W. Kryscinski, S. Yavuz, Y. Liu, X. V. Lin, S. Joty, Y. Zhou, C. Xiong, R. Ying, A. Cohan, and D. Radev. Folio: Natural language reasoning with first-order logic. [Online]. Available: https://arxiv.org/abs/2209.00840

[44] E. Wiewiora, "Potential-based shaping and q-value initialization are equivalent," *Journal of Artificial Intelligence Research*, vol. 19, p. 205–208, Sep. 2003. [Online]. Available: http://dx.doi.org/10.1613/jair.1190

[45] R. Devidze, "Reward design for reinforcement learning agents," 2025. [Online]. Available: https://arxiv.org/abs/2503.21949

[46] A. Doan, A. Halevy, and Z. Ives, "4 - string matching," in *Principles of Data Integration*, A. Doan, A. Halevy, and Z. Ives, Eds. Boston: Morgan Kaufmann, 2012, pp. 95–119. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780124160446000041

[47] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proceedings of the 26th Annual International Conference on Machine Learning*, ser. ICML '09. New York, NY, USA: Association for Computing Machinery, 2009, p. 41–48. [Online]. Available: https://doi.org/10.1145/1553374.1553380

[48] P. Soviany, R. T. Ionescu, P. Rota, and N. Sebe, "Curriculum learning: A survey," 2022. [Online]. Available: https://arxiv.org/abs/2101.10382

[49] T. Matiisen, A. Oliver, T. Cohen, and J. Schulman, "Teacher-student curriculum learning," 2017. [Online]. Available: https://arxiv.org/abs/1707.00183

[50] L. Jiang, D. Meng, Q. Zhao, S. Shan, and A. Hauptmann, "Self-paced curriculum learning," in *No.1: The Twenty-Ninth Conference on Artificial Intelligence Volume*, 01 2015, pp. 2694–2695.

[51] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," 2016. [Online]. Available: https://arxiv.org/abs/1511.05952