MÁSTER EN INGENIERÍA INDUSTRIAL

TRABAJO DE FIN DE MÁSTER

DESARROLLO DE UNA HERRAMIENTA DE VISUALIZACIÓN DE RESULTADOS PARA APLICACIONES DE APRENDIZAJE POR REFUERZO

Autor:

Manuel Barril Rodríguez-Arana

Directores:

Jaime Boal Martín-Larrauri

Lucía Güitta López

Madrid

Junio de 2024

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título

"Desarrollo de una herramienta de visualización de resultados para aplicaciones de aprendizaje por refuerzo"

en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas
en el curso académico 2023/2024 es de mi autoría, original e inédito y
no ha sido presentado con anterioridad a otros efectos.

El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido tomada de otros documentos está debidamente referenciada.

Fdo.: Manuel Barril Rodríguez-Arana Fecha: 20/06/2024

Autorizada la entrega del proyecto

LOS DIRECTORES DEL PROYECTO

Fdo.: Jaime Boal Martín-Larrauri Fecha: …… / …… / ……

Fdo.: Lucía Güitta López Fecha: …25… / …06… / …24…

MÁSTER EN INGENIERÍA INDUSTRIAL

TRABAJO DE FIN DE MÁSTER

DESARROLLO DE UNA HERRAMIENTA DE VISUALIZACIÓN DE RESULTADOS PARA APLICACIONES DE APRENDIZAJE POR REFUERZO

Autor:

Manuel Barril Rodríguez-Arana

Directores:

Jaime Boal Martín-Larrauri

Lucía Güitta López

Madrid

Junio de 2024

Agradecimientos

Quería agradecer a mis directores, Jaime Boal Martín-Larrauri y Lucía Güitta López, cuya visión y conocimiento del tema me han guiado en este proyecto. Sin su apoyo y paciencia no habría podido disfrutar y desarrollar el proyecto.

Me gustaría expresar mi más sincero agradecimiento a mi familia por creer siempre en mí y por su apoyo incondicional.

Desarrollo de una herramienta de visualización de resultados para aplicaciones de aprendizaje por refuerzo

Autor: Barril Rodríguez-Arana, Manuel
Directores: Jaime Boal Martín-Larrauri, Lucía Güitta López
Entidad Colaboradora: Instituto de Investigación Tecnológica (IIT)

RESUMEN DEL PROYECTO

Palabras clave: *machine learning*, aprendizaje por refuerzo, bases de datos, diseño conceptual, React

1. Introducción

La demanda por entender y analizar eficazmente el comportamiento del agente en entornos complejos origina la necesidad de desarrollar una herramienta de visualización para resultados de entrenamiento y evaluación en aprendizaje por refuerzo [1]. La visualización permite a expertos e investigadores identificar patrones, analizar errores, y ajustar estrategias de aprendizaje, fomentando así la innovación y la toma de decisiones. Esto mejora el rendimiento y eficiencia de los algoritmos de aprendizaje por refuerzo, impactando directamente en la calidad y utilidad de las aplicaciones desarrolladas. Adicionalmente, la visualización democratiza el aprendizaje por refuerzo, haciéndolo accesible a un público más amplio, y requiere de herramientas para visualizar intuitivamente los resultados y estrategias de los algoritmos.

2. Definición del proyecto

En este escenario, se ha creado una herramienta de visualización especializada que aborda limitaciones de *frameworks* actuales, enfocada en aprendizaje por refuerzo. Los requisitos para esta herramienta incluyen accesibilidad, personalización, almacenamiento de experimentos históricos, integración transparente y código abierto, facilitando su uso y adaptabilidad según las necesidades específicas del usuario, además de promover la transparencia y colaboración entre la comunidad de desarrolladores.

Tabla 1. Comparativa de la Herramienta propuesta frente a W&B y TensorBoard según los requisitos del proyecto.

	Herramienta	W&B	TensorBoard
Accesibilidad	✓	−	✗
Personalización	✓	✓	✓
Entrenamientos históricos	✓	✓	✗
Integración transparente	✓	✗	✗
Código abierto	✓	✗	✓

3. Descripción de la herramienta

Diseño conceptual de la BBDD

Para diseñar una base de datos relacional (SQL) en un algoritmo de aprendizaje por refuerzo, se han identificado y clasificado las siguientes entidades. Como entidades fuertes son el *Model* y el *Environment* ya que pueden ser identificadas sin necesidad de depender de otra entidad:

- *Model* (Modelo): Representa la estructura usada por el agente para estimar consecuencias de sus acciones (e.g., red neuronal).
- *Environment* (Entorno): El dominio en el cual opera el agente (e.g., juego, simulador).

En cambio, las demás entidades son entidades débiles al depender de *Model* o *Environment* para ser identificadas:

- *Model Hyperparameters*: Parámetros configurables del modelo.
- *Model Hyperparameters Str Value*: Valores de hiperparámetros en formato de cadena.
- *Model Hyperparameters Txt Value*: Valores de hiperparámetros en formato de texto.
- *Environment Characteristics*: Propiedades del entorno.
- *Training*: Proceso de ajuste del modelo y estrategia del agente.
- *Training Hyperparameters*: Parámetros del proceso de entrenamiento.
- *Training Hyperparameters Str Value*: Valores de hiperparámetros de entrenamiento en cadena.
- *Training Hyperparameters Num Value*: Valores numéricos de hiperparámetros de entrenamiento.
- *Sampling*: Selección de estados o acciones del entorno.
- *Gradients*: Derivadas parciales usadas para ajustar el modelo.
- *Weights*: Valores ajustables dentro del modelo.
- *Test*: Evaluación del rendimiento del agente post-entrenamiento.
- *Episode*: Secuencia de interacciones del agente con el entorno.
- *Step*: Una única interacción entre agente y entorno.

Estas entidades permiten estructurar la información necesaria para el entrenamiento y evaluación del agente en el aprendizaje por refuerzo. Las relaciones entre entidades se muestran en el diagrama de la Ilustración 1.

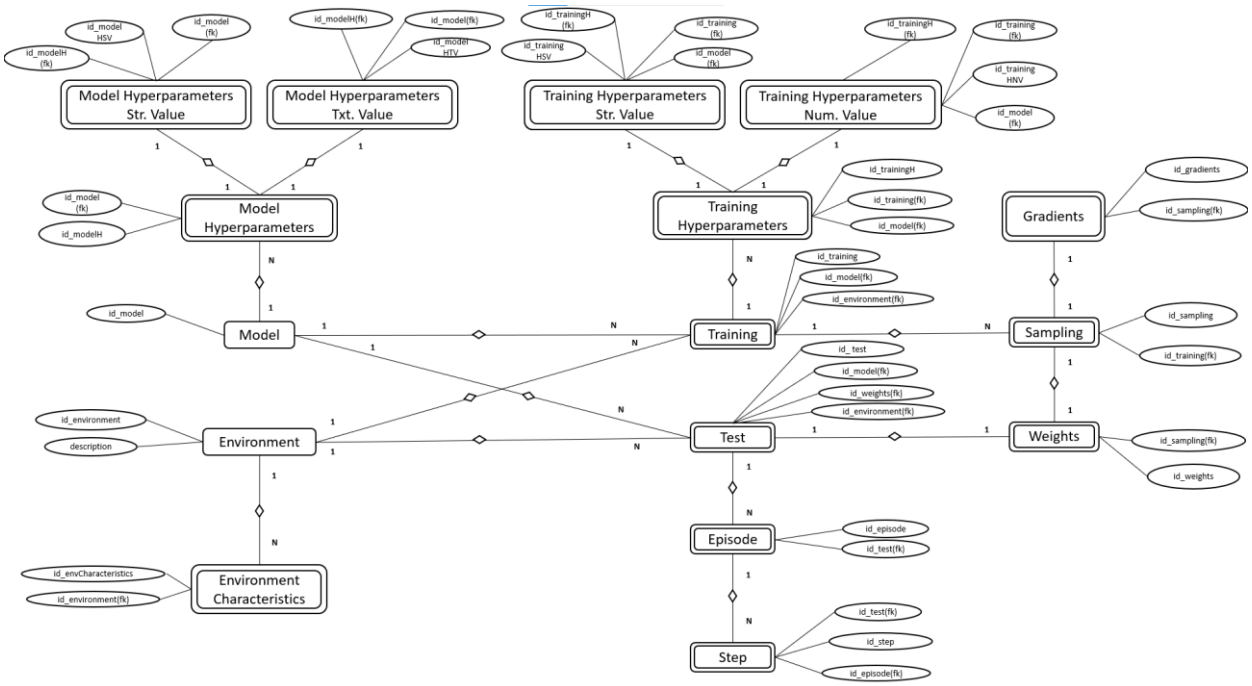


Ilustración 1 Diseño conceptual de un caso genérico de aplicación para aprendizaje por refuerzo.

Backend

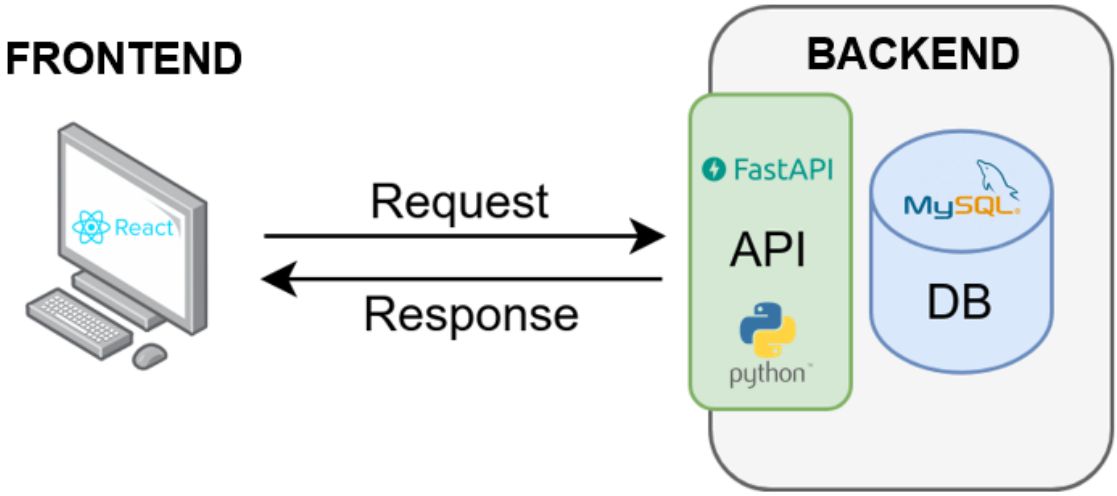


Ilustración 2 Diagrama del frontend y backend de la Herramienta incluyendo tecnologías.

El proyecto consiste en una herramienta con tres capas: presentación (React [2]), lógica de gestión de servicios (API FastAPI [3]) y datos (BBDD MySQL[4]), organizada en servicios independientes que interactúan para formar una aplicación completa. Se implementó el ERM conceptual en un servidor MySQL, utilizando MySQL Workbench para el diseño y visualización de la BBDD. Se definieron entidades y atributos, y se exportó el diagrama EER como una consulta SQL para generar la BBDD en el servidor.

Para la conexión con la BBDD se utilizó Python [5] y la librería mysql.connector, estableciendo una configuración del servidor y utilizando un cursor para ejecutar consultas SQL y recuperar resultados. La comunicación con la BBDD se realiza para cargar y extraer información, permitiendo al usuario subir archivos con datos de entrenamientos y extraer información de entrenamientos específicos.

La API RESTful, creada con FastAPI, facilita la interacción entre el *frontend* y el *backend*, proporcionando *endpoints* para diferentes servicios como la carga de datos y el análisis de entrenamientos. Se gestionan operaciones como la elección de entrenamientos, visualización de resultados de evaluaciones y descarga de datos de entrenamientos. En la carga de datos, se descomprimen archivos zip subidos por el usuario, se procesan y se almacenan en la BBDD. Para el análisis de entrenamientos, se ofrecen opciones al usuario basadas en los datos almacenados para visualizar o descargar información específica.

La integración con el *frontend* se logra mediante *endpoints* asociados a métodos HTTP, permitiendo solicitudes GET y POST para interactuar con la API y acceder a los servicios ofrecidos por el *backend*, facilitando así el intercambio de datos y la realización de diferentes operaciones en la herramienta.

Frontend

El proyecto desarrollado presenta una interfaz de usuario realizada con React, enfocándose en la accesibilidad y facilidad de uso. Se estructura en componentes modulares, facilitando la organización del código y su escalabilidad. Dos funciones principales se destacan: análisis de entrenamientos y carga de datos en la base de datos, cada una con un componente dedicado en React. El componente principal "App" coordina la navegación entre estas secciones.

En el componente de carga de datos, los usuarios pueden seleccionar y cargar archivos zip con datos de entrenamiento en la BBDD. Se proporciona retroalimentación sobre el estado de la carga y se permite la carga de múltiples archivos. En el componente de análisis de entrenamientos, los usuarios pueden seleccionar un modelo y entorno específicos, visualizar un resumen de entrenamientos, seleccionar un ID de entrenamiento para un análisis más detallado, y visualizar gráficos de entrenamiento o descargar pesos del modelo.

Para la comunicación entre el *frontend* y el *backend* se utiliza Axios, una biblioteca de JavaScript para gestionar solicitudes HTTP, asegurando una interacción fluida y segura entre cliente y servidor. Se destaca la modularidad y la estrategia de gestión de estados en React para ofrecer una experiencia de usuario coherente y receptiva, permitiendo análisis detallados y gestión eficaz de los datos de entrenamiento.

4. Caso de uso

En este proyecto se utiliza un entorno virtual con un brazo robótico IRB 120 de ABB para aplicar aprendizaje por refuerzo, con el objetivo de que el robot alcance un cubo rojo en un máximo de pasos por episodio. Se entrenó al agente en episodios, acumulando experiencia en 70 millones de pasos. Cada 50 mil pasos se interrumpe el entrenamiento para evaluar el desempeño a través de 40 episodios de evaluación [6]. Se hizo un diseño conceptual basado en el caso de uso, que incluye entidades y atributos necesarios para guardar datos del entrenamiento. El *backend* implementa este diseño en un diagrama entidad-relación extendido usando MySQL Workbench. En el *frontend*, los usuarios pueden cargar datos de entrenamiento a través de una interfaz que luego permite analizar los entrenamientos,

visualizar resúmenes, gráficos, resultados de *tests*, y descargar pesos de entrenamientos específicos para su posterior análisis o uso. La interfaz proporciona una manera efectiva de interactuar y analizar los entrenamientos de aprendizaje por refuerzo en el contexto de control de un brazo robótico.

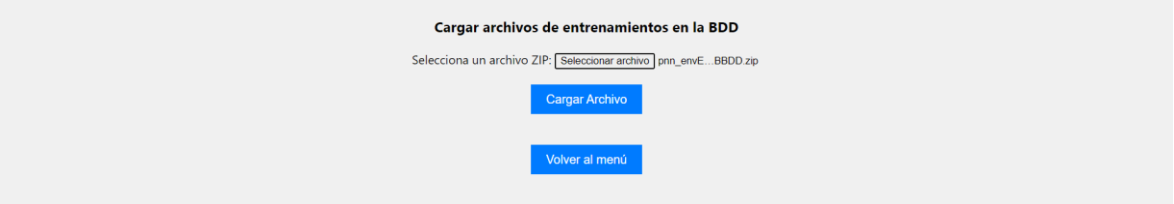


Ilustración 3 Interfaz del usuario en el estado de cargar archivos.

Ilustración 4 Interfaz del usuario en el estado de análisis de entrenamientos.

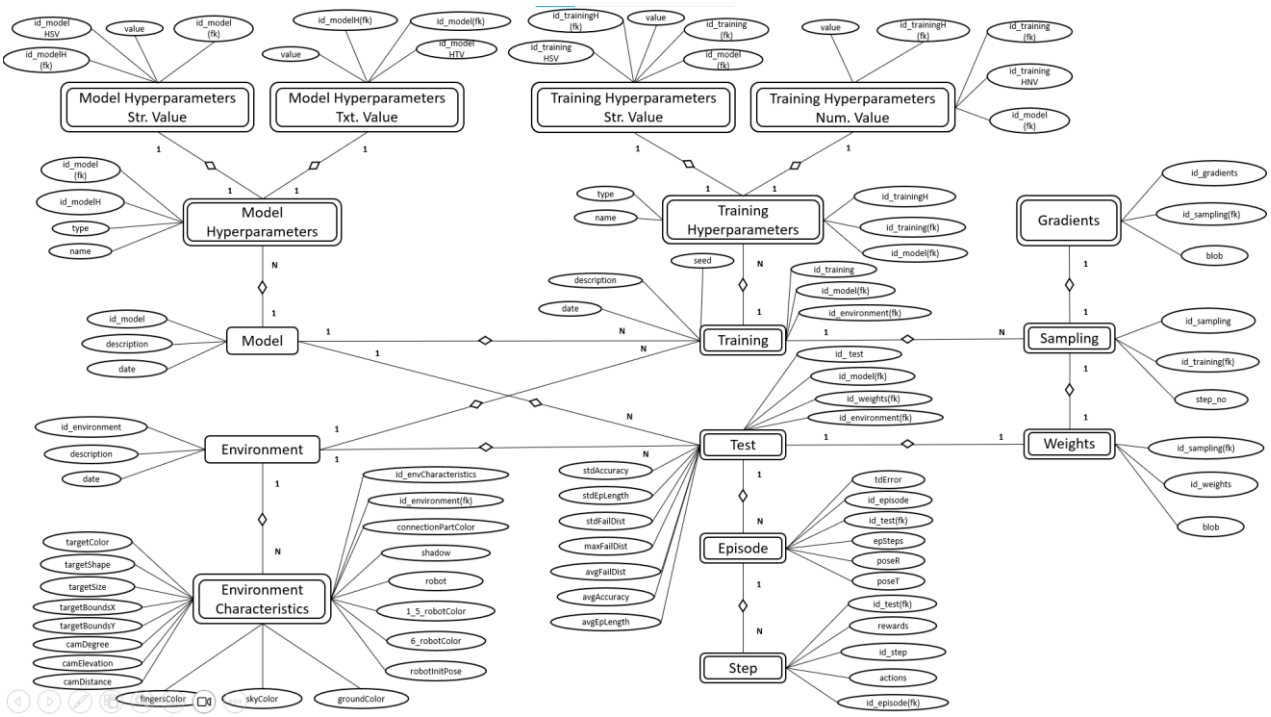


Ilustración 5 Diseño conceptual de un caso de uso con robot.

5. Conclusión y futuro desarrollo

La herramienta desarrollada es funcional y específica para el caso de uso del proyecto, proporcionando soluciones precisas. Su diseño robusto y versátil permite su aplicación a otros casos de aprendizaje por refuerzo. Destaca por su capacidad para cargar 5 GB de datos en aproximadamente 10 minutos y sus transiciones rápidas y fluidas, asegurando un flujo

de trabajo eficiente. Estas características la hacen útil tanto para su caso específico como para una amplia variedad de situaciones en el ámbito del aprendizaje por refuerzo.

La aplicación permite operar localmente con una BBDD en MySQL Workbench, ideal para experimentos de pocas personas involucradas. Sin embargo, se sugiere migrar la BBDD a la nube para favorecer la accesibilidad a equipos más grandes, aunque esto plantea preocupaciones de seguridad que requieren evaluación. Para futuros desarrollos, se podría mejorar la interfaz de usuario, enfocándose en hacerla más intuitiva y estéticamente agradable. Esto facilitaría la interacción y la experiencia del usuario, optimizando la usabilidad del sistema y aumentando la eficiencia en su uso diario.

6. Referencias

[1] Sutton, Richard S., and Andrew G. Barto, "Introduction to reinforcement learning", Vol. 135. Cambridge: MIT press, 1998.

[2] React, React Docs, Accedido Junio 2024; disponible online en: https://legacy.reactjs.org/docs/getting-started.html

[3] FastAPI, FastAPI documentation, Accedido Junio 2024; disponible online en: https://fastapi.tiangolo.com/

[4] MySQL. Accedido Mayo 2022; disponible online en: https://www.mysql.com/

[5] Python, Python documentation, Accedido Junio 2024; disponible online en: https://www.python.org/doc/

[6] Güitta-López, L., Boal, J. & López-López, Á.J. Learning more with the same effort: how randomization improves the robustness of a robotic deep reinforcement learning agent. Appl Intell 53, 14903–14917 (2023). https://doi.org/10.1007/s10489-022-04227-3

Development of a results visualization tool for reinforcement learning applications

Author: Barril Rodríguez-Arana, Manuel
Supervisors: Jaime Boal Martín-Larrauri, Lucía Güitta López
Collaborating Entity: Institute for Research in Technology (IIT)

EXECUTIVE PROJECT SUMMARY

Keywords: machine learning, reinforcement learning, databases, conceptual design, React

1. Introduction

The demand to effectively understand and analyze agent behavior in complex environments gives rise to the need to develop a visualization tool for training and evaluation results in reinforcement learning [1]. Visualization allows experts and researchers to identify patterns, analyze errors, and adjust learning strategies, thus fostering innovation and informed decision-making. This improves the performance and efficiency of reinforcement learning algorithms, directly impacting the quality and usefulness of the developed applications. Additionally, visualization democratizes reinforcement learning, making it accessible to a broader audience,

2. Definition of the project

In this scenario, a specialized visualization tool has been created that addresses limitations of current frameworks, focused on reinforcement learning. The requirements for this tool include accessibility, customization, storage of historical experiments, transparent integration and open source, facilitating its use and adaptability according to the specific needs of the user, in addition to promoting transparency and collaboration among the developer community.

Table 1 Comparison of the proposed tool against W&B and TensorBoard according to the project requirements.

	Tool	W&B	TensorBoard
Accesibility	✓	−	✗
Customization	✓	✓	✓
Historical training records	✓	✓	✗
Seamless integration	✓	✗	✗
Open source	✓	✗	✓

3. Tool description

Conceptual design of the database

To design a relational database (SQL) in a reinforcement learning algorithm, the following entities have been identified and classified. The Model and the Environment are strong entities since they can be identified without having to depend on another entity:

- Model: Represents the structure used by the agent to estimate consequences of its actions (e.g., neural network).
- Environment: The domain in which the agent operates (e.g., game, simulator).

The other entities are weak entities as they depend on Model or Environment to be identified:

- Model Hyperparameters: Configurable model parameters.
- Model Hyperparameters Str Value: Hyperparameter values n string format.
- Model Hyperparameters Txt Value: Hyperparameter values in text format.
- Environment Characteristics: Properties of the environment.
- Training: Agent model and strategy adjustment process.
- Training Hyperparameters: Parameters of the training process.
- Training Hyperparameters Str Value: Chain training hyperparameter values.
- Training Hyperparameters Num Value: Numerical values of training hyperparameters.
- Sampling: Selection of states or actions in the environment.
- Gradients: Partial derivatives used to fit the model.
- Weights: Adjustable values within the model.
- Test: Evaluation of the agent's performance post-training.
- Episode: Sequence of interactions of the agent with the environment.
- Step: A single interaction between agent and environment.

These entities allow structuring the information necessary for the training and evaluation of the agent in reinforcement learning. The relationships between entities are shown in the diagram in Illustration 1.

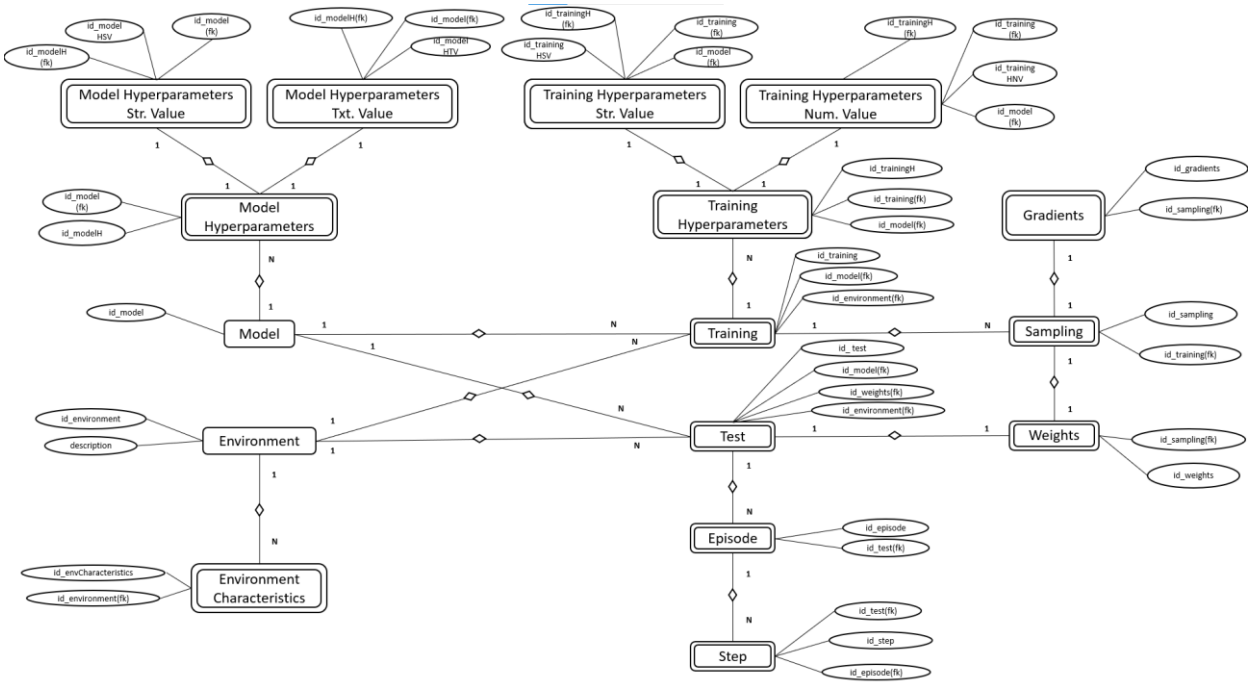


Illustration 1 Conceptual design of a generic application case for reinforcement learning.

Backend

The project consists of a tool with three layers: presentation (React [2]), service management logic (FastAPI API [3]) and data (MySQL database [4]), organized in independent services that interact to form an application complete. The conceptual ERM was implemented on a MySQL server, using MySQL Workbench for the design and visualization of the database. Entities and attributes were defined, and the EER diagram was exported as an SQL query to generate the database on the server.

To connect to the database, Python [5] and the mysql.connector library were used, establishing a server configuration and using a cursor to execute SQL queries and retrieve

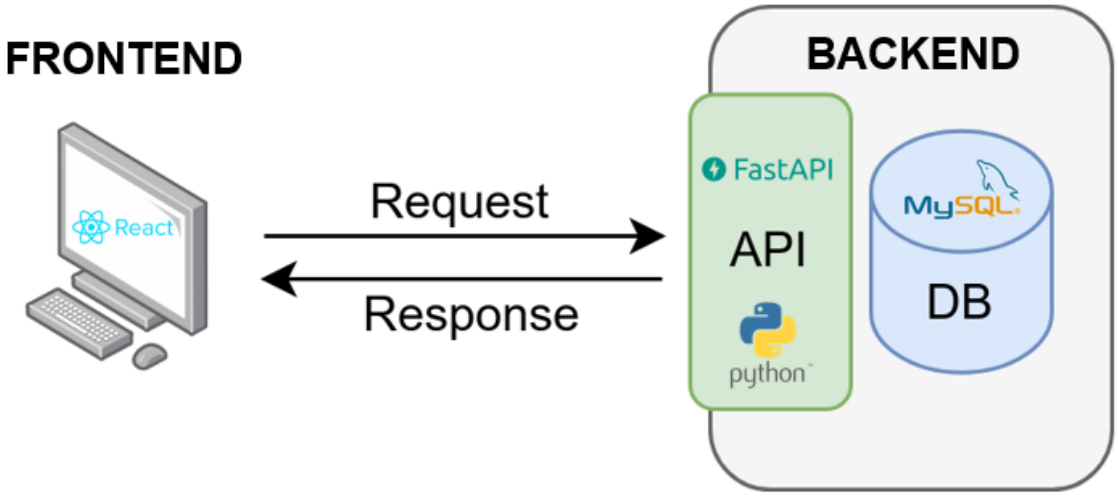


Illustration 2 Diagram of the frontend and backend of the Tool including technologies.

results. Communication with the database is carried out to upload and extract information, allowing the user to upload files with training data and extract information from specific training sessions.

The RESTful API, built with FastAPI, facilitates the interaction between the frontend and backend, providing endpoints for different services such as data loading and training analysis. Operations such as choosing training sessions, viewing evaluation results and downloading training data are managed. When loading data, zip files uploaded by the user are decompressed, processed and stored in the database. For training analysis, the user is offered options based on the stored data to view or download specific information.

Integration with the frontend is achieved through endpoints associated with HTTP methods, allowing GET and POST requests to interact with the API and access the services offered by the backend, thus facilitating the exchange of data and performing different operations in the tool.

Frontend

The developed project presents a user interface made with React, focusing on accessibility and ease of use. It is structured in modular components, facilitating the organization of the code and its scalability. Two main functions stand out: training analysis and loading data into the database, each with a dedicated component in React. The main "App" component coordinates navigation between these sections.

In the data upload component, users can select and upload zip files with training data to the database. Feedback on upload status is provided and multiple file uploads are allowed. In the training analysis component, users can select a specific model and environment, view a training summary, select a training ID for further analysis, and view training graphs or download model weights.

For communication between the frontend and the backend, Axios is used, a JavaScript library to manage HTTP requests, ensuring a fluid and secure interaction between client and server. The modularity and state management strategy in React is highlighted to deliver a consistent and responsive user experience, enabling detailed analysis and effective management of training data.

4. Use case

This project uses a virtual environment with an ABB IRB 120 robotic arm to apply reinforcement learning, with the goal of the robot reaching a red cube in a maximum of steps per episode. The agent was trained in episodes, accumulating experience in 70 million steps. Every 50 thousand steps the training was interrupted to evaluate performance through 40 evaluation episodes [6]. A conceptual design was designed based on the use case, which includes entities and attributes necessary to store training data. The backend implements this layout in an extended entity-relationship diagram using MySQL Workbench. On the frontend, users can upload training data through an interface that then allows them to analyze the training, view summaries, graphs, test results, and download weights from specific workouts for later analysis or use. The interface provides an effective way to

interact and analyze reinforcement learning training in the context of controlling a robotic arm.

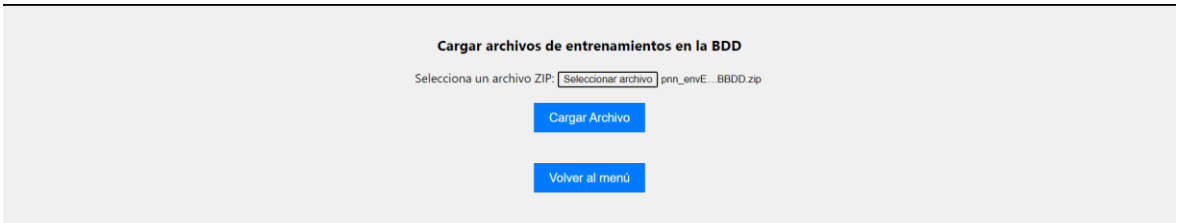


Illustration 3 User interface in file upload state.

Illustration 4 User interface in training analysis status.

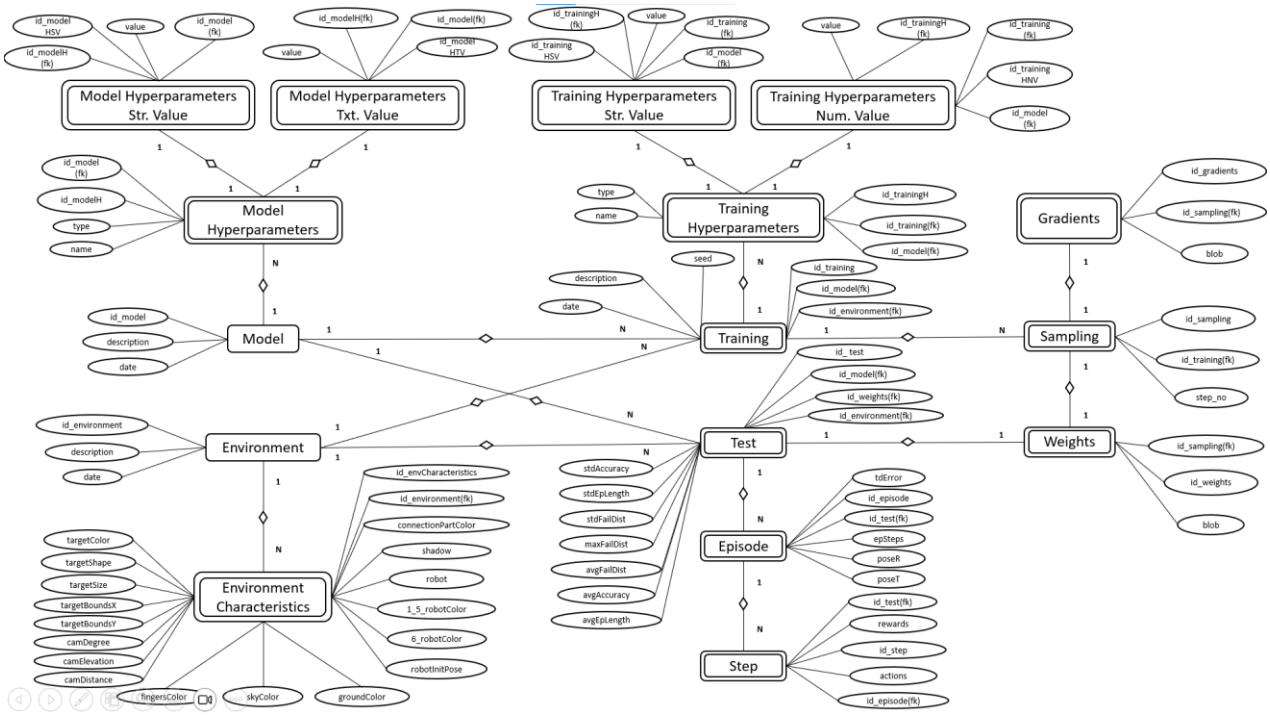


Illustration 5 Conceptual design of a use case with a robot.

5. Conclusion and future development

The developed tool is functional and specific to the project's use case, providing precise solutions. Its robust and versatile design allows for application to other reinforcement learning scenarios. It stands out for its ability to load 5 GB of data in approximately 10 minutes and its swift, seamless transitions, ensuring efficient workflow. These features make it valuable for both its specific use case and a wide range of reinforcement learning situations.

The application allows for local operation with a MySQL Workbench database, ideal for experiments involving a small number of people. However, migrating the database to the cloud is recommended to enhance accessibility for larger teams, though security concerns need careful evaluation. For future developments, improving the user interface to make it more intuitive and aesthetically pleasing could be beneficial. This would enhance user interaction and experience, optimizing system usability and increasing efficiency in daily use.

6. References

[1] Sutton, Richard S., and Andrew G.Barto, “Introduction to reinforcement learning”, Vol. 135. Cambridge: MIT press, 1998.

[2] React, React Docs, Accessed June 2024; available online at:https://legacy.reactjs.org/docs/getting-started.html

[3] FastAPI, FastAPI documentation, Accessed June 2024; available online at:https://fastapi.tiangolo.com/

[4] MySQL. Accessed June 2022; available online at:https://www.mysql.com/

[5] Python, Python documentation, Accessed June 2024; available online at:https://www.python.org/doc/

[6] Güitta-López, L., Boal, J. & López-López, Á.J. Learning more with the same effort: how randomization improves the robustness of a robotic deep reinforcement learning agent. Appl Intell 53, 14903–14917 (2023). https://doi.org/10.1007/s10489-022-04227-3

Índice de la memoria

Capítulo 1. Introducción 5

Aprendizaje automático 5

Motivación 6

Objetivos 6

Capítulo 2. Estado de la cuestión 8

Herramientas de visualización 8

Weights & Biases 8

TensorBoard 9

Comparativa Weights & Biases y Tensorboard 10

Frameworks para aplicaciones web 10

Interfaz de programación de aplicaciones 11

Bases de datos 13

Capítulo 3. Diseño conceptual de la base de datos 14

Diagrama conceptual ERM 14

Entidades y atributos 15

Relaciones 18

Capítulo 4. Backend 20

Implementación de la base de datos 20

Diseño conceptual ERM aplicado a un servidor MySQL 20

Conexión de la base de datos 23

Interfaz de programación de aplicaciones 23

Carga de datos 24

Análisis de los entrenamientos 26

Integración con frontend 28

Capítulo 5. Frontend 29

Componentes de la página web en React 29

Componente principal 30

Componente cargarDatos.js 31

Componente analisisEntrenamientos.js 32

Integración con backend a través de AXIOS 34

Capítulo 6. Caso de uso 35

Descripción 35

Base de datos 36

Backend 38

Frontend 40

Capítulo 7. Conclusión y futuro desarrollo 42

BBDD con servidor en la nube 42

Mejora del diseño de la interfaz de usuario 43

Bibliografía 44

Anexo ODS 46

Índice de figuras

Figura 1 Diseño conceptual de un caso genérico de aplicación para aprendizaje por refuerzo 14
Figura 2 Diagrama del frontend y backend de la Herramienta incluyendo tecnologías..... 20
Figura 4 Creación base de datos con script SQL en MySQL Workbench 22
Figura 3 Diagrama EER de la BBDD generalizada 22
Figura 5 Diagrama de proceso de la funcionalidad de carga de datos. 26
Figura 6 Función encargada de añadir atributos para nuevas gráficas. 26
Figura 7 Diagrama de proceso de la funcionalidad de análisis de los entrenamientos 26
Figura 8 Endpoints y métodos utilizados para desarrollar la aplicación. 28
Figura 9 Interfaz de usuario en la página principal. 30
Figura 10 Diseño conceptual de un caso de uso con robot..... 37
Figura 11 Diagrama EER de la BBDD para el caso de uso con robot. 38
Figura 15 Interfaz del usuario cuando la carga de datos en la BBDD ha finalizado. 39
Figura 14 Interfaz del usuario cuando la carga de datos en la BBDD está en proceso. 39
Figura 13 Interfaz del usuario en el estado de cargar archivos. 39
Figura 12 Ejemplo de muestra de valores de la BBDD del caso de uso con robot mediante queries de SQL para la tabla de la entidad Model Hyperparameters Str. Value en la izquierda y Environment Characteristics en la derecha. 39
Figura 16 Interfaz del usuario en el estado de análisis de entrenamientos. 39
Figura 18 Interfaz de usuario en la elección del entrenamiento. 40
Figura 17 Interfaz del usuario mostrando los entrenamientos que coinciden con la combinación de modelo y entorno escogidas. 40
Figura 20 Interfaz de usuario en el proceso de descarga de los pesos del entrenamiento seleccionado..... 41
Figura 21 Interfaz de usuario con en el proceso de descarga de los pesos del entrenamiento seleccionado finalizado..... 41
Figura 22 Interfaz de usuario mostrando los resultados del test seleccionado. 41
Figura 19 Interfaz de usuario mostrando las gráficas del entrenamiento seleccionado. 41

Índice de tablas

Tabla 1. Comparativa Herramienta frente a W&B y TensorBoard según requisitos del proyecto 7
Tabla 2 Comparativa W&B y Tensorboard según requisitos del proyecto 10
Tabla 3 Características de Angular, React y Vue.js 11
Tabla 4 Características de FastAPI, Django y Flask 12
Tabla 5 Entidades y atributos generelalizados para una aplicación de aprendizaje por refuerzo 18
Tabla 6 Entidades y atributos para caso de uso con robot 36

Capítulo 1. Introducción

Aprendizaje automático

El aprendizaje automático, una rama de la inteligencia artificial, ha revolucionado la forma en que se abordan los problemas y presenta grandes ventajas en comparación con los enfoques algorítmicos más tradicionales. Estos algoritmos de aprendizaje automático tienen la capacidad de extraer características y patrones útiles de los datos de manera automática, sin requerir una programación manual intensiva. Esto permite a los modelos actualizarse y adaptarse a medida que se reciben nuevos datos, lo que los hace altamente escalables. Dentro del aprendizaje automático, se pueden identificar diferentes enfoques, entre los que destacan el aprendizaje supervisado, el aprendizaje no supervisado y el aprendizaje por refuerzo.

En el aprendizaje supervisado [1] se deben proporcionar los datos de entrada al algoritmo, así como la salida que se espera. El algoritmo busca identificar patrones y establecer una relación entre las entradas y las salidas esperadas.

Por otro lado, el aprendizaje no supervisado [2] se enfoca en encontrar patrones ocultos o estructuras intrínsecas en los datos. El algoritmo busca agrupar los datos en categorías o descubrir características comunes entre ellos. Este enfoque es especialmente útil en la exploración de grandes conjuntos de datos, ya que puede revelar información valiosa y desconocida sobre los datos sin la necesidad de la supervisión del operador.

El aprendizaje por refuerzo [3] se basa en la interacción entre un agente y su entorno. El agente toma la decisión de qué acción realizar en el entorno, y recibe retroalimentación en forma de una recompensa según el resultado de la acción. El objetivo del agente es aprender a tomar decisiones óptimas que maximicen las recompensas a largo del episodio. A través de la retroalimentación continua, el agente ajusta su comportamiento y desarrolla estrategias que le permiten enfrentar eficientemente situaciones dinámicas y cambiantes.

Motivación

La motivación para desarrollar una herramienta de visualización de los resultados de entrenamientos y de la evaluación de un algoritmo de aprendizaje por refuerzo radica en la necesidad de comprender y analizar de manera efectiva el proceso de aprendizaje y el comportamiento del agente en entornos complejos. A medida que la tecnología avanza y el aprendizaje automático adquiere cada vez más importancia en diversos campos, se vuelve crucial contar con herramientas que permitan visualizar y entender de manera intuitiva los resultados y las estrategias desarrolladas por los algoritmos de aprendizaje por refuerzo.

La visualización no solo contribuye a la innovación tecnológica, sino que también facilita la toma de decisiones informadas. Al proporcionar una representación gráfica y comprensible del proceso de aprendizaje, los expertos y los investigadores pueden identificar patrones, analizar errores y fallos, y ajustar las estrategias de aprendizaje de manera más efectiva. Esto a su vez permite mejorar el rendimiento y la eficiencia de los algoritmos, lo que tiene un impacto directo en la calidad y la utilidad de las aplicaciones y sistemas desarrollados.

Además, la visualización ayuda a democratizar el campo del aprendizaje automático al hacerlo más accesible y comprensible para un público más amplio. Al presentar los resultados y las estrategias en formatos visuales e intuitivos, se fomenta una comprensión más amplia y profunda de los conceptos y procesos involucrados en el aprendizaje automático.

Objetivos

En este contexto, se ha desarrollado una herramienta de visualización centrada en la funcionalidad para apoyo en los entrenamientos de algoritmos de aprendizaje por refuerzo, que tiene como objetivo abordar algunas de las limitaciones que presentan los *frameworks* de visualización actuales. Esta herramienta se centra en la incorporación de una base de datos específica para el modelo de aprendizaje por refuerzo. Los requisitos exigidos a esta herramienta son los siguientes:

- Accesibilidad: La herramienta debe ser accesible y comprensible para usuarios de diversas necesidades según su campo de trabajo.
- Personalización: Debe permitir la personalización de su funcionamiento y apariencia para adaptarse a las necesidades específicas del usuario, enfocado siempre a modelos de aprendizaje por refuerzo.
- Almacenamiento de experimentos históricos completos: Debe ser capaz de almacenar y gestionar registros históricos de experimentos, lo que implica un sistema de gestión de datos sólido y eficiente.
- Integración transparente: No debe ser necesario el modificar el código para que funcione correctamente y tiene que ser versátil en cuanto a las gráficas que se pueden visualizar.
- Código abierto (*open source*): Debe ser un software de código abierto, lo que significa que su código fuente es accesible y modificable por la comunidad de desarrolladores, lo que fomenta la transparencia y la colaboración.

Según los requisitos establecidos en los objetivos del proyecto, la Tabla 1 muestra como la herramienta desarrollada en este proyecto, a partir de ahora "Herramienta", se busca que se presente como una mejor opción frente a las alternativas actuales que son Weights & Biases y TensorBoard.

Tabla 1. Comparativa de la Herramienta propuesta frente a W&B y TensorBoard según los requisitos del proyecto.

	Herramienta	W&B	TensorBoard
Accesibilidad	✓	−	✗
Personalización	✓	✓	✓
Entrenamientos históricos	✓	✓	✗
Integración transparente	✓	✗	✗
Código abierto	✓	✗	✓

Capítulo 2. Estado de la cuestión

Herramientas de visualización

Tras la continua transformación y desarrollo de los algoritmos de aprendizaje automático, se ha generado un creciente interés en el desarrollo de herramientas de visualización que faciliten la comprensión y análisis de los algoritmos de aprendizaje, tanto de lo sucedido durante el entrenamiento como a posteriori en las evaluaciones. En el caso del aprendizaje por refuerzo, estas herramientas tienen como objetivo proporcionar representaciones gráficas del entorno, mostrar las políticas y estrategias aprendidas por el agente, hacer un seguimiento del progreso del aprendizaje, analizar errores y fallos, e incluso permitir la interacción en tiempo real con el agente y el entorno.

Entre las herramientas de visualización más utilizadas en el campo del aprendizaje por refuerzo se encuentran Weights & Biases [4] y TensorBoard [5]. Herramientas que en sí tienen ventajas destacadas, pero que, como se detalla en los objetivos, en este proyecto no cumplen con todos los requisitos exigidos.

Weights & Biases

Weights & Biases (W&B) es una plataforma utilizada para el seguimiento y visualización de experimentos en el campo del aprendizaje automático. W&B permite al usuario las siguientes opciones:

- Seguimiento de experimentos: Permite visualizar métricas en tiempo real de un entrenamiento que está activo.
- Visualización: Proporciona herramientas de visualización para comparar entrenamientos y estudiar el modelo obtenido.
- Registro de artefactos: Permite guardar información sobre el conjunto de datos utilizado, modelo y cualquier artefacto que se relaciona con los entrenamientos.

- Colaboración: Destaca por permitir la colaboración entre miembros de un proyecto y compartir información sobre los entrenamientos y resultados de manera sencilla.
- Integración: Se integra con una amplia variedad de *frameworks* de aprendizaje automático, como PyTorch [6], Keras [7] o TensorFlow [8].

Para utilizar W&B se debe realizar la instalación de la biblioteca, iniciar sesión y posteriormente integrar con el modelo. Hace falta introducir en el código del entrenamiento los comandos de la librería para que de esta manera se puedan monitorizar las métricas.

Weights & Biases es una herramienta que aporta un gran valor en proyectos de aprendizaje automático en los que participan un grupo de personas. Ya que al guardar los datos de los códigos, configuraciones y resultados presenta una gran oportunidad de colaboración, comunicación y reproductibilidad.

TensorBoard

TensorBoard [8] es una herramienta de visualización originalmente creada para TensorFlow, y con el tiempo se ha vuelto más accesible y se puede utilizar con varios *frameworks* de aprendizaje profundo como PyTorch o Keras. Permite comprender y visualizar flujos de trabajo de entrenamientos. Destacan las siguientes funcionalidades de la herramienta:

- Monitorear y observar indicadores como el error y la exactitud.
- Ilustrar el diagrama del modelo, incluyendo las operaciones y capas.
- Observar gráficos de distribución de pesos, sesgos y otros tensores conforme evolucionan en el tiempo.
- Visualizar incorporaciones en un espacio de dimensiones reducidas.
- Exhibir imágenes, datos de texto y de sonido.
- Analizar perfiles de aplicaciones.

Para utilizar TensorBoard se necesita instalar el paquete y una vez se tenga, se añade al código desarrollado para el modelo creado. La información generada por la ejecución del modelo se carga en un directorio de registro en el dispositivo local. En ese directorio accede

el *framework* para poder realizar las visualizaciones y demás funciones anteriormente mencionadas.

Comparativa Weights & Biases y Tensorboard

Se muestra en la Tabla 2 una comparativa de W&B y Tensorboard en las características buscadas en este proyecto.

Tabla 2 Comparativa W&B y Tensorboard según requisitos del proyecto

	W&B	TensorBoard
Accesibilidad	⊖	✗
Customización	✓	✓
Entrenamientos históricos	✓	✗
Integración transparente	✗	✗
Código abierto	✗	✓

Frameworks para aplicaciones web

La interacción entre el software y el usuario se lleva a cabo mediante interfaces implementadas en una amplia gama de aplicaciones, tanto en el ámbito de las aplicaciones de escritorio como en las aplicaciones web, que son las más utilizadas en la actualidad en términos de visualización de datos. Las aplicaciones web permiten implementar interfaces de usuario a través de Internet, lo que proporciona acceso desde cualquier dispositivo con conexión a la web.

En el ámbito de las aplicaciones web, los desarrolladores cuentan con una variedad de herramientas y *frameworks* para facilitar el desarrollo y mantenimiento de interfaces de usuario. Entre los *frameworks* más populares utilizados en la actualidad se encuentran Angular [9], Vue [10] y React [11], cada uno con sus propias características y ventajas.

Angular y React son *frameworks* mantenidos por grandes empresas multinacionales, como Google y Meta, respectivamente. Estos *frameworks* ofrecen un conjunto completo de soluciones para las necesidades de los desarrolladores de aplicaciones web. Angular, por ejemplo, se destaca por su capacidad para ofrecer validaciones de formularios, envíos de

solicitudes HTTP y enrutamiento, entre otras características incluidas. React, por otro lado, se centra en ofrecer herramientas y enfoques para el desarrollo de interfaces de usuario, brindando una mayor flexibilidad y permitiendo un desarrollo más completo en comparación con los otros *frameworks*. En cuanto a Vue, este *framework* es gestionado por un equipo de colaboradores distribuidos en todo el mundo. Si bien Vue ofrece menos funciones que Angular, tiene más características que React. Vue se enfoca en proporcionar una experiencia intuitiva y fácil de usar para los desarrolladores, permitiendo una curva de aprendizaje más suave y una sintaxis clara y concisa.

En resumen, tanto Angular, Vue y React son *frameworks* populares para el desarrollo de interfaces de usuario en aplicaciones web. Cada uno tiene sus fortalezas y se adapta a diferentes necesidades. Angular ofrece una amplia gama de características integradas, React se centra en el desarrollo de interfaces de usuario y Vue ofrece una experiencia intuitiva y fácil de usar. Las ventajas quedan resumidas en la Tabla 3 [12] [13] [14] [15].

Tabla 3 Características de Angular, React y Vue.js

	ANGULAR	REACT	VUE.JS
Creado por	Google®	Meta®	Vue.js
Popularidad	⊖	✓	⊖
Curva de aprendizaje	✗	✓	✓
Comunidad	✓	✓	✗

Interfaz de programación de aplicaciones

Cuando el usuario interactúa con una página web, se produce un intercambio de información entre la página y la base de datos. Este proceso de comunicación es gestionado por una interfaz de programación de aplicaciones (API). En el contexto de este proyecto, se ha elegido el lenguaje de programación Python para desarrollar la API debido a su gran versatilidad y la amplia variedad de aplicaciones que ofrece, así como su sólida colección de bibliotecas y una gran comunidad de desarrolladores que facilita el soporte y el desarrollo continuo.

En Python [16], existen varios *frameworks* ampliamente utilizados para el desarrollo de APIs. Algunos de los más destacados son Django [17], FastAPI [18] y Flask [19]. Cada uno de estos *frameworks* tiene sus características que los hacen adecuados para diferentes tipos de proyectos y requisitos.

FastAPI presenta un eficiente rendimiento y capacidad para manejar grandes volúmenes de solicitudes. Se destaca por su enfoque en la documentación automática y la validación de datos. Además, FastAPI se integra fácilmente con bibliotecas populares de Python, lo que facilita la construcción de sistemas complejos y escalables.

Django, por otro lado, es un *framework* web completo y modular que ofrece una amplia gama de características integradas para el desarrollo de aplicaciones. Es conocido por su enfoque en la productividad y la escalabilidad. Django ofrece una asignación automática de objetos de un programa a una base de datos relacional y viceversa, una sólida capa de autenticación y una interfaz de administración fácil de usar. Además, cuenta con una gran comunidad de desarrolladores y una amplia gama de paquetes y complementos disponibles, lo que facilita el desarrollo de aplicaciones web robustas y personalizadas.

Por último, Flask es un *framework* web minimalista y flexible que se enfoca en la simplicidad y la extensibilidad. Proporciona una base sólida para construir aplicaciones web y permite una gran libertad en cuanto a la estructura y la implementación del código. Cabe destacar que Flask se caracteriza por una suave curva de aprendizaje.

En la Tabla 4 se comparan FastAPI, Django y Flask con las principales características que se buscan para este proyecto.

Tabla 4 Características de FastAPI, Django y Flask

	FastAPI	Django	Flask
Velocidad (performance)	✔	⊖	⊖
Adecuado para cualquier tamaño	✔	✔	⊖
Flexibilidad del código	✔	⊖	✔

Bases de datos

La información proveniente del modelo entrenado por aprendizaje por refuerzo requiere de una estructura de almacenamiento para garantizar la trazabilidad y accesibilidad de los datos. En este contexto, se utilizan bases de datos que se adaptan a las necesidades específicas de los datos y del sistema.

Las bases de datos con un lenguaje de consultas estructurados (SQL) son ampliamente utilizadas y se caracterizan por tener un esquema definido y una estructura tabular. Utilizan un lenguaje de consultas estructurado que permite realizar operaciones como inserción, modificación y consulta de datos de manera eficiente. Estas bases de datos son útiles cuando se requiere un alto nivel de integridad y coherencia en los datos, así como para aplicaciones que siguen una estructura bien definida.

Por otro lado, las bases de datos con un lenguaje de consulta no estructurado (NoSQL) son una opción flexible que se adapta a diversos escenarios. A diferencia de las bases de datos SQL, las bases de datos NoSQL no tienen un esquema fijo, lo que las hace más adecuadas para datos no estructurados. Pueden ser orientadas a documentos, grafos, columnas u otros formatos, y ofrecen una mayor escalabilidad horizontal al permitir distribuir los datos en múltiples servidores. Además, son útiles para aplicaciones con requisitos de alto rendimiento y escalabilidad, ya que pueden manejar grandes volúmenes de datos y operaciones concurrentes de manera eficiente.

Es importante tener en cuenta que la elección entre una base de datos SQL o NoSQL dependerá de los requisitos específicos del proyecto. Si la estructura de los datos es rígida y se necesita garantizar la integridad y coherencia de estos, una base de datos SQL podría ser la opción adecuada. Por otro lado, si los datos son menos estructurados o se requiere una mayor escalabilidad y flexibilidad, las bases de datos NoSQL ofrecen una solución más versátil.

Capítulo 3. Diseño conceptual de la base de datos

Las bases de datos (BBDD) son el conjunto de datos organizados que se gestionan de forma estructurada con el objetivo de ser accedidos de manera eficiente. El diseño de la BBDD es único para cada tipo de aplicación. En este proyecto la aplicación a estudiar son los algoritmos de aprendizaje por refuerzo, donde los datos generados por los resultados de los entrenamientos presentan una estructura rígida y que se repite en el tiempo. Se toma la decisión de escoger una BBDD con lenguaje estructurado (SQL) debido a las características estructurales de los datos a estudiar. La base de datos debe capturar información detallada sobre los estados, acciones, recompensas, y las transiciones entre estados, así como los parámetros y resultados del modelo de aprendizaje. A continuación, se presenta un diseño detallado de las entidades y sus atributos, enfocado en soportar de manera robusta y eficiente el proceso de aprendizaje por refuerzo.

Diagrama conceptual ERM

Las entidades, atributos y relaciones han quedado definidas para el caso general de aplicaciones de aprendizaje por refuerzo. El diagrama final se muestra en la Figura 1.

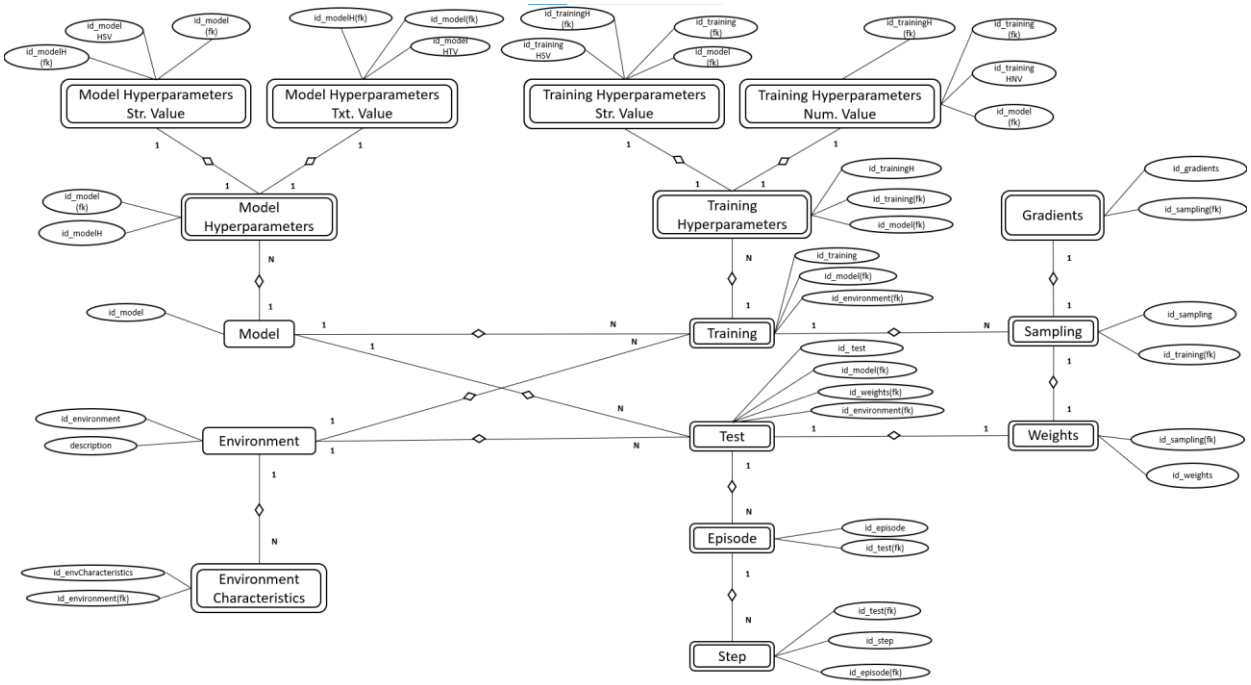


Figura 1 Diseño conceptual de un caso genérico de aplicación para aprendizaje por refuerzo.

Entidades y atributos

Una vez se ha decidido que se utiliza una BBDD de tipo SQL se procede a definir cuáles son las entidades. Se define entidad a los conceptos del entrenamiento del algoritmo por refuerzo del que se desea mantener la información. Estas entidades tienen asociadas atributos que son las características o propiedades que definen una entidad específica.

Las entidades según sus características propias y de lo que representan se pueden clasificar en entidades fuertes y débiles. Una entidad fuerte es una entidad de la BBDD que es independiente y que puede existir por sí misma. La entidad que tenga esa dependencia se considera une entidad débil, pudiendo ser débil por existencia o por identificación.

Son entidades fuertes en el caso de estudio el *Model* y el *Environment* ya que pueden ser identificadas sin necesidad de depender de otra entidad:

- *Model* (Modelo): El modelo en un algoritmo de aprendizaje por refuerzo se refiere a la representación que el agente utiliza para estimar las consecuencias de sus acciones en el entorno. Puede tomar varias formas, como una red neuronal o una tabla de valores.
- *Environment* (Entorno): El entorno es el dominio en el que el agente de aprendizaje por refuerzo opera. Puede ser un juego, un simulador físico o cualquier otro espacio donde el agente tome decisiones.

En cambio, las demás entidades son entidades débiles al depender de *Model* o *Environment* para ser identificadas:

- *Model Hyperparameters* (Hiperparámetros del Modelo): Estos son los parámetros configurables que afectan directamente al modelo y que no se ajustan durante el entrenamiento.
- *Model Hyperparameters Str Value* (Valor de Cadena de los Hiperparámetros del Modelo): Esta entidad se refiere a los valores de los hiperparámetros del modelo expresados en formato de cadena.

- *Model Hyperparameters Txt Value* (Valor de Texto de los Hiperparámetros del Modelo): Aquí se encuentran los valores de los hiperparámetros del modelo expresados en formato de tipo texto.
- *Environment Characteristics* (Características del Entorno): Estas son las propiedades o rasgos clave del entorno que afectan la interacción del agente con él.
- *Training* (Entrenamiento): El entrenamiento se refiere al proceso mediante el cual el agente de aprendizaje por refuerzo ajusta su modelo y estrategia de toma de decisiones para maximizar las recompensas a lo largo del tiempo. En esta entidad se van a crear atributos que corresponden con las posibles gráficas que el usuario quiera mostrar. Dependiendo de las gráficas que quiera guardar se añadirán atributos.
- *Training Hyperparameters* (Hiperparámetros de Entrenamiento): Son los parámetros configurables que afectan el proceso de entrenamiento, como la cantidad de episodios o la frecuencia de actualización del modelo.
- *Training Hyperparameters Str Value* (Valor de Cadena de los Hiperparámetros de Entrenamiento): Estos son los valores de los hiperparámetros de entrenamiento expresados en formato de tipo cadena.
- *Training Hyperparameters Num Value* (Valor Numérico de los Hiperparámetros de Entrenamiento): Estos son los valores de los hiperparámetros de entrenamiento expresados en formato de tipo numérico.
- *Sampling* (Muestreo): El muestreo se refiere al proceso de seleccionar estados o acciones del entorno para poder realizar un seguimiento de los entrenamientos.
- *Gradients* (Gradientes): En el contexto de un algoritmo de aprendizaje profundo, los gradientes son las derivadas parciales de la función de pérdida con respecto a los parámetros del modelo. Ayudan a ajustar los parámetros durante el entrenamiento.
- *Weights* (Pesos): Los pesos son los valores ajustables dentro del modelo, como los pesos de una red neuronal. Son actualizados durante el entrenamiento para mejorar el rendimiento del modelo.
- *Test* (Prueba): En esta entidad, se realizan pruebas y evaluaciones para medir el rendimiento final del agente después del entrenamiento.

- *Episode* (Episodio): Un episodio en el aprendizaje por refuerzo es una secuencia completa de interacciones del agente con el entorno, desde un estado inicial hasta un estado final o terminal.
- *Step* (Paso): Un paso se refiere a una única interacción entre el agente y el entorno, donde el agente toma una acción en un estado y recibe una recompensa.

Los atributos [20] se pueden clasificar según composición, valores u origen. Por composición pueden ser simples cuando presentan un solo componente o compuestos cuando presentan varios. Según los valores que presenten pueden ser monovaluados cuando existe un valor por cada ocurrencia o multivaluados cuando existen varios valores para cada ocurrencia. Por origen pueden ser almacenados cuando se almacenan físicamente en la BBDD y derivado cuando el valor del atributo se puede calcular a partir de otros atributos. En el caso de estudio se han obtenido todos los atributos como atributos simples, monovaluados y almacenados.

De todos los atributos que presenta una entidad existen un atributo o atributos que sirven como identificadores de dicha entidad. Según su función de identificadores se clasifican en claves primarias, claves foráneas o atributos simples. Las claves primarias son los atributos no nulos que permiten identificar de manera única cada fila de una tabla y garantiza que no haya ningún duplicado en la tabla. Una clave foránea es un atributo de una entidad que a su vez en otra entidad corresponde a una clave primaria. Los atributos simples son características de la entidad que la describen en mayor profundidad. Debido a que en este caso se está tratando una generalización de los entrenamientos de aprendizaje por refuerzo, los atributos simples son omitidos ya que dependen de cada caso específico.

En la Tabla 5 se resumen las entidades y atributos para algoritmos de aprendizaje por refuerzo incluyendo la distinción por atributos de claves primarias, claves foráneas y omitiendo los atributos simples (en el Capítulo 6 se muestra un caso de uso con sus atributos simples).

Tabla 5 Entidades y atributos generalizados para una aplicación de aprendizaje por refuerzo

Entidad	**Atributo**	
	Clave primaria	**Clave foránea**
Model	id_model	-
Model Hyperparameters	id_modelH	id_model
Model Hyperparameters Str Value	id_modelHSV	id_model, id_modelH
Model Hyperparameters Txt Value	id_modelHTV	id_model, id_modelH
Environment	id_environment	-
Environment Characteristics	id_envCharacteristics	id_environment
Training	id_training	id_environment, id_model
Training Hyperparameters	id_trainingH	id_training, id_model
Training Hyperparameters Str Value	id_trainingHSV	id_training, id_model, id_trainingH
Training Hyperparameters Num Value	id_trainingHNV	id_training, id_model, id_trainingH
Sampling	id_sampling	id_training
Gradients	id_gradients	id_sampling
Weights	id_weights	id_sampling
Test	id_test	id_model, id_weights, id_environment, id_test
Episode	id_episode	id_test
Step	id_test	id_step, id_episode

Con las entidades ya escogidas se crea un modelo entidad-relación (ERM). Las entidades se consideran tablas, los atributos son atributos de las tablas y las relaciones entre entidades son identificativas o no identificativas.

Relaciones

En un ERM las relaciones son las conexiones entre diferentes entre las entidades. Presentan una gran importancia en el modo en el que las entidades se van a relacionar y como se

intercambia la información dentro de la base de datos. La cardinalidad de la relación se expresa en el diagrama de ERM en términos de "uno a uno" (1:1), "uno a muchos" (1:N) y "mucho a muchos" (N:N). Donde:

- 1:1 representa que un registro en una entidad A se relaciona con un registro de otra entidad B, y viceversa. En el caso de estudio un ejemplo serían los hiperparámetros del modelo (*Model Hyperparameters*) que se relacionan únicamente con un registro de cadena (*Model Hyperparameters Str Value*) o un registro numérico (*Model Hyperparameters Num Value*), ya que es una característica del propio hiperparámetro y solo puede ir asociado a sí mismo.
- 1:N representa que un registro en una entidad A se relaciona con varios registros de una entidad B, pero un registro de B se relaciona con solo un registro de A. En el caso de estudio un ejemplo serían los modelos (*Model*) que cada uno presenta varios hiperparámetros (*Model Hyperparameters*) que lo definen. Esos hiperparámetros solo pueden definir a un modelo únicamente.
- N:N representa que un registro en una entidad A se relaciona con varios registros de una entidad B, pero un registro de B se relaciona con varios registros de A. En el caso de estudio no se ha considerado la utilización de este tipo de relaciones debido a que ninguna relación entre entidades las cumple.

Un modelo se define a través de varios hiperparámetros (1:N). Un hiperparámetro del modelo puede ser de tipo cadena o de tipo texto (1:1). Un modelo tiene varios entrenamientos asociados (1:N) que a su vez ese entrenamiento tiene varios hiperparámetros asociados (1:N) que pueden ser de tipo cadena o de tipo numérico (1:1). Cada entrenamiento está compuesto por distintas muestras (1:N) y que a su vez tienen asociado un gradiente propio (1:1) y unos pesos (1:1). Un modelo presenta varias pruebas (1:N) y estas pruebas están asociados a unos resultados de pesos (1:1). Las pruebas están compuestas por varios episodios (1:N) que a su vez están compuestos por varios pasos (1:N).

La lógica para la entidad de entorno es muy similar. Un entorno se compone de varias características (1:N). Un entorno tiene varios entrenamientos y pruebas asociadas (1:N).

Capítulo 4. Backend

La Herramienta creada en este proyecto es una aplicación que presenta tres capas que consiste en una capa de presentación (la página de React), una capa de lógica de gestión de servicios (la API FastAPI) y una capa de datos (la BBDD MySQL). Tanto la API como la BBDD son las capas que forman el backend de la aplicación. La estructura que se ha seguido para favorecer la organización y eficiencia, donde diferentes servicios independientes, cada uno con su propia función específica, interactúan entre sí para formar una aplicación completa.

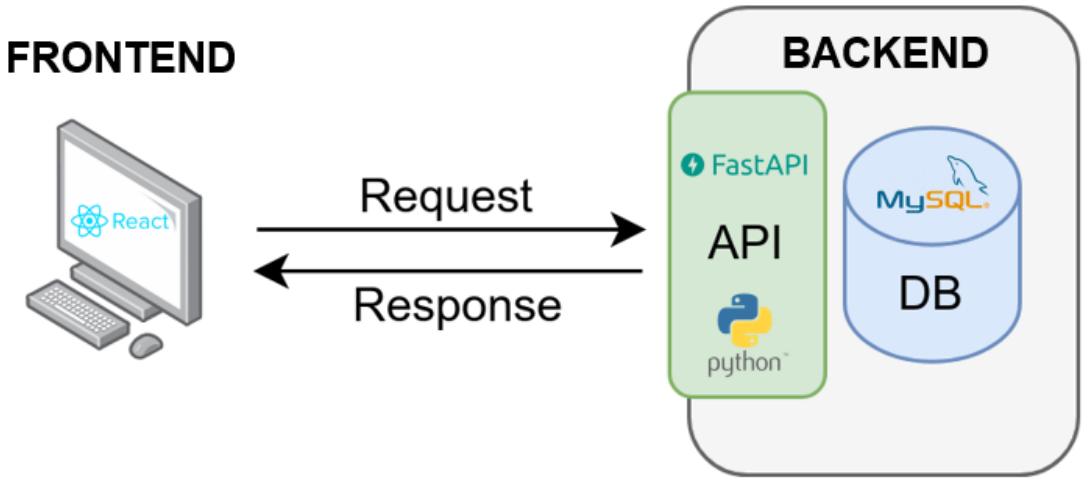


Figura 2 Diagrama del frontend y backend de la Herramienta incluyendo tecnologías.

Implementación de la base de datos

Diseño conceptual ERM aplicado a un servidor MySQL

Se procede a implementar el ERM en un servidor para que pueda ser de utilidad a la hora de crear la BBDD. Se ha escogido MySQL como gestor de bases de datos (DBMS) debido a que es intuitivo, flexible, gratis, de código abierto y presenta una gran variedad de herramientas disponibles para aprovechar la gestión de base de datos como es MySQL Workbench. MySQL Workbench facilita tanto el diseño de la BBDD como la visualización

de datos. Por un lado, permite diseñar y crear esquemas de bases de datos utilizando una interfaz gráfica. Por otro lado, proporciona herramientas de visualización de datos que permiten explorar y comprender mejor el contenido de la base de datos a través de gráficos y diagramas.

Se utilizan los diagramas entidad-relación extendido (EER) que presenta MySQL Workbench para poder trasladar el ERM conceptual a un diagrama funcional. En la Figura 3 se muestra parte del diagrama creado para este proyecto. Esta herramienta visual facilita que se trasvase toda la información del modelo conceptual debido a que presenta una gran similitud con la estructura del diseño conceptual.

Se crean las entidades y se asignan los atributos correspondientes indicando que atributo es clave primaria en la entidad. Es necesario indicar que tipo de dato tiene cada atributo del diagrama. Para ello MySQL ofrece diferentes tipos de datos soportados. En este caso se utilizan de tipo:

- *Int*: Para almacenar números enteros.
- *Longtext*: Para almacenar textos largos.
- *Datetime*: Para almacenar fechas y horas.
- *Varchar* (45): Para almacenar campos de texto de longitud variables con longitud máxima de 45 caracteres.
- *Mediumblob*: Para almacenar datos binarios.

El diagrama EER final se exporta como una consulta SQL para poder utilizarlo desde cualquier gestor de BBDD de SQL. La consulta se ejecuta en MySQL para poder genera la base de datos en el servidor, en este caso local. En la Figura 4 se muestra un ejemplo de la creación de la base de datos llamada “ciclab” en el servidor local del ordenador.

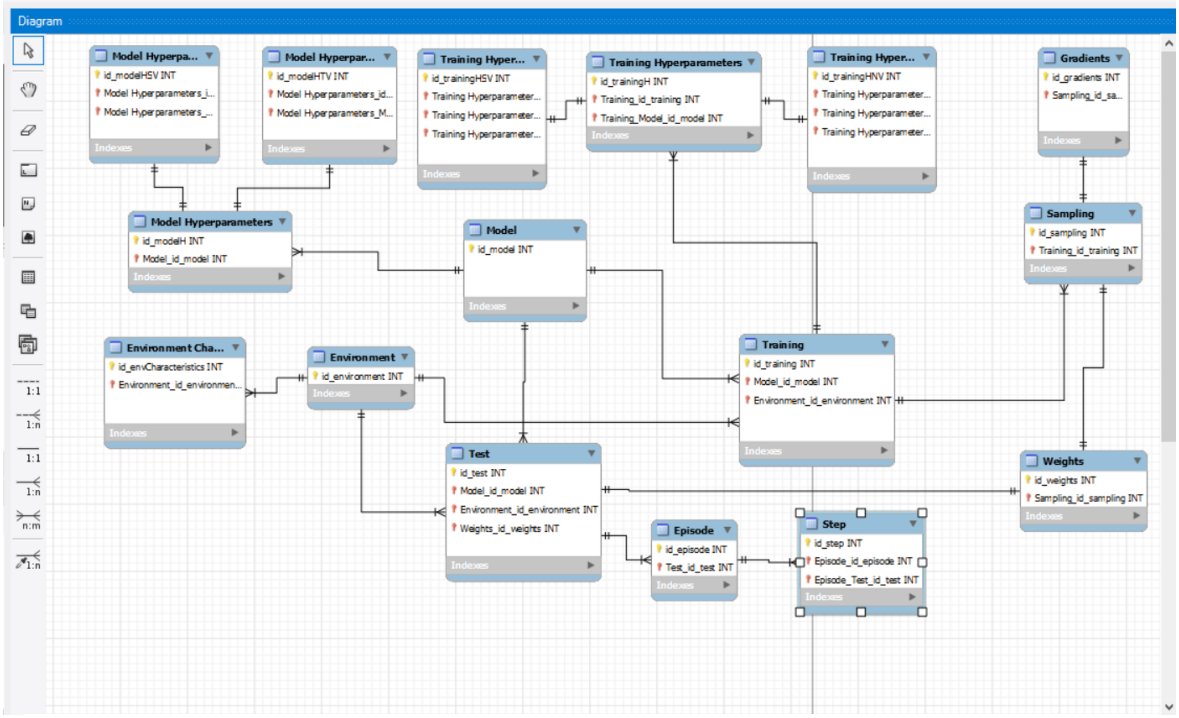


Figura 3 Diagrama EER de la BBDD generalizada.

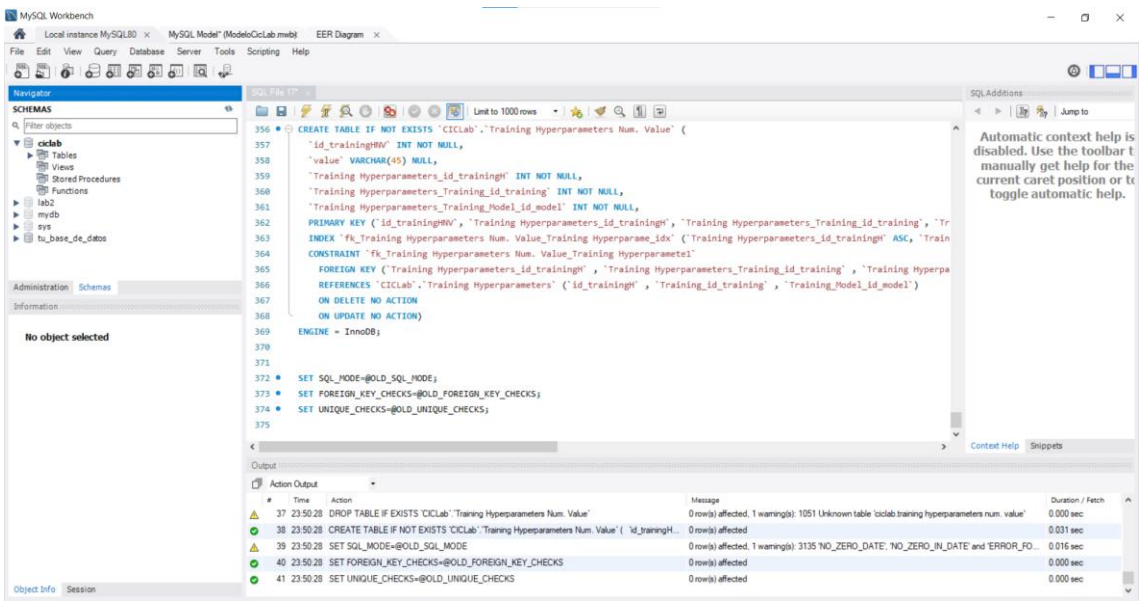


Figura 4 Creación base de datos con script SQL en MySQL Workbench.

Conexión de la base de datos

La BBDD se encuentra en un servidor al cual se establece la conexión a través del lenguaje de programación Python. Se utiliza la librería de Python mysql.connector para conseguir desde un entorno de desarrollo, en este caso se utilizó Visual Studio Code, la conexión con el servidor MySQL que contiene la BBDD.

El proceso que se ha seguido en esta parte del código comienza con la creación de la configuración del servidor de la BBDD. Esta configuración se basa en los parámetros de host, usuario, contraseña y el nombre de la BBDD. Posteriormente se crea un cursor que actúa como un objeto con el que se permite ejecutar consultas SQL. El cursor sirve de punto de acceso para ejecutar consultas SQL y recuperar resultados de la BBDD. Una vez que se ha establecido la conexión con la BBDD, se almacenan los resultados para poder ser utilizados en otros pasos del proceso. Después de haber completado las consultas y haber procesado los resultados, se finaliza la operación cerrando tanto el cursor como la conexión con la BBDD. Esto se hace para liberar recursos, preservar la integridad de la BBDD, prevenir bloqueos y facilitar el reciclaje de conexiones.

La comunicación con la BBDD se realiza para cargar información y para extraer información. Por un lado, la interfaz del usuario se comunicará con la BBDD a través de la metodología antes mencionada y permitirá al usuario poder subir archivos para guardar sus datos en la BBDD. Por otro lado, el usuario pedirá la información de un cierto entrenamiento y para ello se extraerá esa información, se procesará y se mostrará en el *frontend*.

Interfaz de programación de aplicaciones

Una API (Interfaz de Programación de Aplicaciones, por sus siglas en inglés) es un conjunto de reglas y protocolos que permiten que diferentes aplicaciones se comuniquen entre sí. En el contexto de desarrollo web, una API generalmente se refiere a una interfaz que permite que una aplicación o servicio proporcione datos o funcionalidades a otras aplicaciones, ya sea de forma interna en un sistema o a través de la web para que las aplicaciones externas las utilicen.

La página web desarrollada con React interactúa con una API en el *backend* para obtener datos y funcionalidades adicionales. La conexión entre el *frontend* (la página web React) y el *backend* (API y BBDD) sirve para poder intercambiar los datos de la BBDD entre ambas partes ya sea para cargarlos o extraerlos.

En este proyecto se crea una aplicación de API RESTful utilizando el *framework* FastAPI en Python. FastAPI es una herramienta moderna y eficiente que facilita la construcción de APIs en Python de manera rápida, siguiendo el principio de REST (*Representational State Transfer*). Este tipo de API permite una interacción estructurada entre el cliente y el servidor, utilizando operaciones estándar como GET, POST, PUT o DELETE sobre recursos identificables a través de *URLs*. La API RESTful ha sido empleada en el *backend* de una aplicación web desarrollada con React, proporcionando una interfaz de comunicación que maneja las solicitudes y respuestas de datos entre el *frontend* y el servidor. En este escenario, React se encarga de la interfaz de usuario en el navegador web, mientras que FastAPI procesa las solicitudes provenientes del *frontend*, permitiendo un flujo de datos entre las diferentes partes de la aplicación. Esta combinación resulta en una arquitectura robusta y escalable, que facilita la implementación de funcionalidades complejas y una respuesta rápida ante las interacciones del usuario.

El código de la API se ha dividido en dos servicios según las funcionalidades ofrecidos por la aplicación desarrollada. Por un lado, la carga a la BBDD y por otro lado el análisis de entrenamientos que presenta las siguientes funciones: la elección del entrenamiento, mostrar gráficas de los entrenamientos, mostrar resultados de evaluaciones y poder descargar en local los datos de entrenamientos específicos. En la carga de archivos se realizan modificaciones de carga de datos en la BBDD y en el análisis de entrenamientos no se hacen modificaciones de la BBDD ya que solo interactúa con ella para procesos de extracción.

Carga de datos

Para la carga de los entrenamientos a la BBDD se recibe un archivo zip, elegido para facilitar la transferencia y ahorro de espacio de los archivos, por parte del *frontend* subido por el usuario. En ese archivo zip se guardan todos los resultados y las métricas de los

entrenamientos de la aplicación de aprendizaje por refuerzo. El archivo viene en formato zip y se descomprime en el *backend*. Esa carpeta queda guardada en el local y desde ahí se recorren las carpetas y se cargan los valores en la BBDD.

El código tiene indicado los nombres de carpetas que tiene que buscar y que datos extraer, por lo que es fundamental mantener dicha estructura para conseguir el correcto funcionamiento del código. El tiempo de subida de archivos según las pruebas realizadas es entorno a los 2 minutos por cada GB.

En la carpeta se encuentran archivos tipo *.pth* y *.pickle* que contienen los datos de los tensores relacionados con los pesos y gradientes. Se van a transformar en formato binario para poder ser cargados en la BBDD debido a la limitación de tipos de datos que se pueden subir en la BBDD de MySQL. Se leen también los archivos relacionados con el modelo y entorno de un archivo Excel. Por cada entrenamiento se guardarán archivos de contenido HTML que contienen las gráficas de análisis y que se guardan en la BBDD como *LONGTEXT*. De forma dinámica se creará un atributo en la entidad training para guardar el contenido HTML, esto permite poder guardar todas las gráficas independientemente de que tenga antes de la carga el atributo estuviera asignado. Se comparan los nombres de los archivos HTML de la carpeta zip subida y de los atributos de la entidad training, si existe un nombre de la carpeta que no estuviera en la BBDD se crea con el código mostrado en la Figura 6. En el caso de que hubiera un atributo designado para guardar HTML y que no coincidiera con los nombres de los archivos HTML de la carpeta zip subida entonces se guarda como vacío en la BBDD, y solo se mostrarán por la página de React los que contengan un valor que no sea vacío. Tras terminar de cargar los datos se elimina el archivo del dispositivo local ya que todo su contenido se encuentra subido a la BBDD.

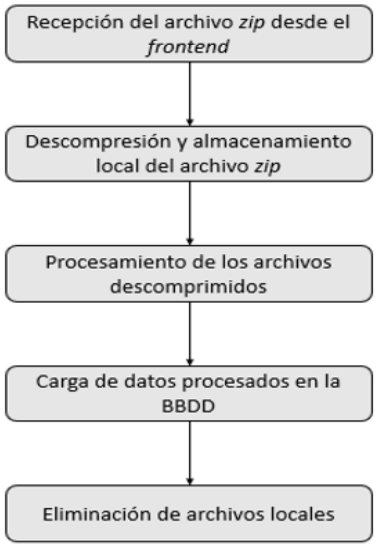


Figura 5 Diagrama de proceso de la funcionalidad de

```python
def add_column_if_not_exists(column_name, cursor, db_connection):
    """Añade una columna a la tabla 'Training' si no existe."""
    alter_query = f"ALTER TABLE CICLab.Training ADD {column_name} LONGTEXT NULL;"
    cursor.execute(alter_query)
    db_connection.commit()
```

Figura 6 Función encargada de añadir atributos para nuevas gráficas.

Análisis de los entrenamientos

Esta funcionalidad se describe a alto nivel en el diagrama de proceso de la figura 7.

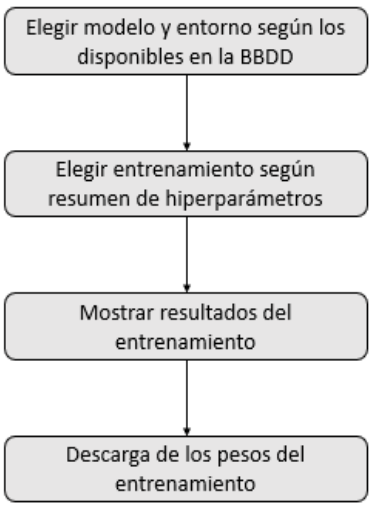


Figura 7 Diagrama de proceso de la funcionalidad de análisis de

Elección del entrenamiento

Se mandan las descripciones de los modelos y entornos que están subidos en la BBDD sin tener en cuenta los valores que se repiten. Se reciben las descripciones de entorno y modelo del *frontend* que describirán a varios entrenamientos posibles ya que pueden existir múltiples entrenamientos asociados a la misma combinación de modelo y entorno. En el *backend* se buscan los valores de los hiperparámetros asociados a cada entrenamiento para mostrarlos y para que el usuario pueda elegir el entrenamiento que busca según esos hiperparámetros. Una vez se le muestran las opciones el usuario elige el entrenamiento que quiere sus gráficas, los resultados de sus evaluaciones o descargar los pesos de ese entrenamiento.

Gestión de los resultados de las evaluaciones

En cada entrenamiento existen unas muestras para poder entender con profundidad como ha sido el proceso del entrenamiento. Esas muestras tienen asociado un número de paso (*step*) para indicar cuando fueron realizadas y además se les realiza una evaluación para poder entender el progreso de los resultados. Con el entrenamiento elegido si el usuario decide que quiere ver los resultados de las evaluaciones de dicho entrenamiento, se acceden a todas las muestras y se obtienen los números de *step* en los que ha ocurrido. El usuario entonces envía al *backend* la elección de que test quiere ver en función del número de *step*. En el *backend* se obtienen mediante consultas los resultados de las evaluaciones que tienen asociado ese número de *step*.

Descarga de entrenamientos

Como se ha explicado anteriormente todos los datos de los entrenamientos quedan registrados en la BBDD y se elimina del local la carpeta original donde se guardaban los datos. Es por ello para lo que se ha creado la funcionalidad de poder descargar información de la BBDD y que el usuario pueda tener una carpeta con información del entrenamiento. Se ha decidido por su importancia que los pesos del modelo se pudieran descargar. El usuario una vez elige el entrenamiento tiene la opción de descargar todos los pesos del modelo asociados al entrenamiento elegido. En el *backend* se recibe el entrenamiento y se accede a los pesos que se encuentran en tipo binario. Estos archivos se transforman en un archivo con tensores de tipo *.pt* que podrá encontrar el usuario en las descargas locales de su ordenador.

Integración con frontend

La integración del *backend* con el *framework* de React se consigue a través de una serie de *endpoints* a los que el *frontend* puede hacer solicitudes. Cada *endpoint* tiene una *URL* única y está asociado con uno de los métodos HTTP (*Hypertext Transfer Protocol*) solicitudes incluyendo métodos como GET, POST, PUT o DELETE. En este proyecto solo se han utilizado los métodos GET y POST debido a las características propias del código que no requerían de otros métodos. En la Figura 8 se muestran los *endpoints* y métodos que se han utilizado.

```
# Ruta para obtener las descripciones de los modelos y entornos
@app.get('/descripciones_modelos_entornos')
async def obtener_descripciones():···

# Ruta para guardar modelo y entorno y obtener datos de training
@app.post('/guardar_modelo_entorno')
async def guardar_modelo_entorno(modelo_entorno: ModeloEntornoIngresado):···

# Ruta para guardar el id del entrenamiento y mostrar las gráficas
@app.post('/guardar_id_entrenamiento')
async def guardar_id_entrenamiento(training_id: TrainingID):···

# Ruta para descargar los weights del entrenamiento
@app.get("/descargar_weights_entrenamiento/{training_id}")
async def descargar_weights_entrenamiento(training_id: str):···

# Ruta para subir archivo con datos del entrenamiento
@app.post("/subir_archivo/")
async def subir_archivo(archivo: UploadFile):···

# Ruta para obtener los step numbers del sampling
@app.get("/obtener_samplings/{id_training}", response_model=List[int])
async def obtener_samplings(id_training: int):···

# Ruta para obtener los resultados del test
@app.get("/obtener_resultados_test/{training_id}/{step_no}")
async def obtener_resultados_test(training_id: int, step_no: int):···
```

Figura 8 Endpoints y métodos utilizados para desarrollar la aplicación.

Capítulo 5. Frontend

El *frontend* representa el primer punto de contacto y única área de interacción con la aplicación. En este proyecto, se ha dado prioridad a la accesibilidad, eliminando la necesidad de que los usuarios realicen ajustes de código. Se han aplicado prácticas de accesibilidad desde el diseño inicial, como la etiquetación adecuada de elementos, la provisión de texto explicativo en los estados y la navegación sencilla a través de botones y desplegables. También se ha prestado atención a la legibilidad, la elección de colores y fuentes accesibles. Cabe destacar que se ha priorizado la funcionalidad frente a la calidad del diseño de visualización.

La interfaz de usuario se realiza a través de React. Esta herramienta es una biblioteca de JavaScript utilizada para construir interfaces de usuario interactivas y dinámicas en aplicaciones web. Fue desarrollada por Meta y se ha convertido en una de las herramientas más populares para la creación de aplicaciones web modernas. Se seleccionó como la principal tecnología para desarrollar la página web del *frontend* debido a su amplia popularidad, curva de aprendizaje accesible y la sólida comunidad de apoyo, tal como se ha observado en el análisis del estado actual del proyecto. React se enfoca en la creación de componentes reutilizables que representan partes específicas de la interfaz de usuario de una aplicación. Los componentes son bloques de construcción que encapsulan la lógica y la interfaz de usuario de una parte específica de la aplicación. Estos componentes pueden ser combinados para formar interfaces de usuario más complejas. En este proyecto se crea una página web de una sola página (SPA) que presenta varias funcionalidades.

Componentes de la página web en React

La estrategia modular es parte de este proyecto, ya que cada sección se ha dividido en componentes individuales. Este enfoque ofrece ventajas significativas al facilitar la organización del código, fomentar la reutilización de componentes y permitir una escalabilidad futura sin complicaciones. Cada componente se ha diseñado para realizar una

función específica dentro de la herramienta, lo que simplifica el mantenimiento y la depuración del sistema.

En este proyecto, se destacan dos funciones principales: el análisis de entrenamientos y la carga de datos en la base de datos. Por esta razón, se ha optado por dividir la arquitectura del código React en tres componentes. Uno de estos componentes se enfoca en el análisis de entrenamientos, mientras que el otro se dedica a la carga de datos en la base de datos. Además, se ha mantenido un tercer componente, que es el componente principal o raíz. Este componente principal coordina la estructura general de la aplicación, organiza la navegación entre los componentes de análisis y carga de datos, y puede gestionar el estado global de la aplicación, si es necesario.

Componente principal

Este componente de React, llamado "App", desempeña el papel de la página principal de la aplicación web. Su función principal es actuar como el punto de entrada y coordinador principal de la aplicación, controlando la interfaz de usuario y la navegación entre dos secciones clave de la aplicación: el "Análisis de entrenamientos" y la "Carga de datos en la base de datos."

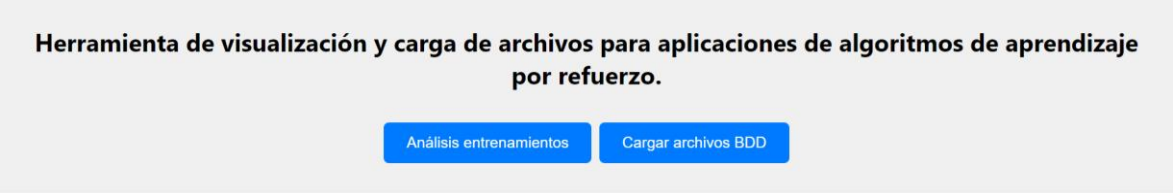


Figura 9 Interfaz de usuario en la página principal.

La interfaz comienza con un encabezado que muestra un título descriptivo de la aplicación. Luego, se presentan dos botones en la pantalla: "Análisis de entrenamientos" y "Carga de archivos BBDD." Estos botones permiten al usuario elegir entre dos funcionalidades principales.

Cuando el usuario hace clic en "Análisis de entrenamientos," se activa la vista correspondiente, que muestra una serie de opciones relacionadas con el análisis de datos de

entrenamiento de algoritmos de aprendizaje por refuerzo. Esta vista la genera el componente asociado de analisisEntrenamientos.js.

Por otro lado, si el usuario hace clic en "Carga de archivos BBDD," se activa la vista de carga de datos a través del componente de cargarDatos.js. Aquí, el usuario puede seleccionar y cargar archivos en la base de datos de la aplicación, lo que está relacionado con la alimentación de datos para análisis posteriores.

La navegación entre estas dos secciones se gestiona mediante la actualización de la una variable que contiene el estado del componente, lo que permite al usuario volver a la página principal y navegar entre las vistas de "Análisis de entrenamientos" y "Carga de datos en la base de datos" de manera fluida.

En resumen, este componente "App" sirve como el punto de inicio y control central de una aplicación web, facilitando al usuario la elección y navegación entre dos funcionalidades principales: el análisis de entrenamientos y la carga de datos en la base de datos.

Componente cargarDatos.js

El componente cargarDatos.js permite a los usuarios cargar archivos de entrenamiento en la BBDD. Se describen sus funcionalidades en secuencia de operación:

- Selección de archivo zip: Los usuarios tienen la capacidad de seleccionar un archivo zip desde su dispositivo local. Esto se logra a través de un elemento de entrada de archivo (`<input type="file">`) que permite navegar y seleccionar el archivo deseado. Véase la Figura 13.
- Carga de archivo a la BBDD: Una vez que el usuario ha seleccionado un archivo zip, pueden iniciar el proceso de carga haciendo clic en el botón "Cargar Archivo", Cuando se hace clic en este botón, el archivo seleccionado se carga en la base de datos de la aplicación a través de una solicitud HTTP utilizando la biblioteca Axios. La carga del archivo se realiza en segundo plano sin requerir una recarga de la página, lo que proporciona una experiencia de usuario más fluida. Véase la Figura 13.

- Estado de carga: Durante el proceso de carga, se muestra un mensaje de "Cargando..." para informar al usuario que la operación está en curso. Esto ayuda a mantener al usuario informado sobre el progreso de la carga. Véase la Figura 14.
- Tiempo de carga estimado: Se proporciona una estimación del tiempo de carga aproximado que depende del tipo de entrenamiento de aprendizaje por refuerzo utilizado. Esto es útil para que los usuarios tengan una idea de cuánto tiempo pueden esperar que dure el proceso de carga. Véase que en la Figura 14 es 10 minutos debido a que se utiliza el valor obtenido en el caso de uso posteriormente desarrollado.
- Confirmación de carga exitosa: Una vez que la carga del archivo se completa con éxito, se muestra un mensaje de confirmación en verde indicando que el archivo se ha cargado correctamente en la BBDD. Véase la Figura 15.
- Cargar más archivos: Para mayor conveniencia, los usuarios tienen la opción de cargar más archivos después de una carga exitosa. Al hacer clic en el botón "Cargar más archivos," se reinician los estados relevantes, lo que permite al usuario seleccionar y cargar otro archivo zip sin salir de esta vista. Véase la Figura 15.
- Volver al menú principal: En cualquier momento, los usuarios pueden regresar al menú principal de la aplicación haciendo clic en el botón "Volver al menú". Véase la Figura 13.

En resumen, este componente permite a los usuarios cargar archivos de entrenamiento en la base de datos de manera eficiente y ofrece una experiencia de usuario informada y conveniente al mostrar mensajes de estado claros y opciones para cargar múltiples archivos si es necesario.

Componente analisisEntrenamientos.js

El componente analisisEntrenamientos.js permite a los usuarios realizar un análisis detallado de los entrenamientos de modelos en diversos entornos. Presenta las siguientes funcionalidades en secuencia de operación:

- Selección de modelo y entorno: Los usuarios tienen la opción de seleccionar un modelo específico y un entorno particular a través de listas desplegables. En estos

desplegables se muestran todas las opciones de modelos y entornos subidos a la BBDD. Esto establece el contexto necesario para el análisis. Véase la Figura 16.

- Envío de datos al servidor: Una vez que los usuarios han seleccionado un modelo y un entorno, pueden enviar estos datos al servidor haciendo clic en el botón "Enviar." Esto desencadena una solicitud al servidor que ejecuta el análisis en segundo plano.
- Resumen de entrenamientos: Después de enviar los datos, se muestra un resumen de los entrenamientos realizados en forma de tabla. Este resumen incluye información relevante sobre los hiperparámetros utilizados en cada entrenamiento. Los usuarios podrán elegir el entrenamiento según los valores de los hiperparámetros. Véase la Figura 17.
- Selección de ID de entrenamiento: Los usuarios tienen la capacidad de seleccionar un ID de entrenamiento específico de una lista desplegable. Esto habilita un análisis más detallado de un entrenamiento específico. Se dan tres opciones de análisis: la visualización de gráficas, los resultados de los *tests* o poder descargar los pesos del entrenamiento seleccionado. Véase Figura la 18.
- Visualización de gráficas de entrenamiento: Una vez que se ha seleccionado un ID de entrenamiento, se pueden mostrar gráficas de entrenamiento relacionadas con ese ID. Estas gráficas se presentan en *iframes* y proporcionan información importante sobre el rendimiento del modelo. Los *iframes* son elementos de tipo HTML que se utilizan para incrustar otro documento HTML dentro de una página web. Se utiliza Plotly [21] como biblioteca de visualización de datos de JavaScript ya que permite generar gráficos interactivos y visualizaciones de datos personalizadas. Véase Figura 19.
- Mostrar *tests* realizados: Los usuarios tienen la opción de ver los *tests* realizados en el modelo seleccionado haciendo clic en el botón "Mostrar Tests." Esto brinda información adicional sobre el rendimiento del modelo en escenarios específicos. Se muestran los números de *step* en los que se han hecho las evaluaciones. Después de seleccionar un *test* específico, se muestran los resultados de ese *test* en una tabla. Estos resultados incluyen información como la precisión, la longitud del episodio, la distancia de falla, entre otros. Véase la Figura 22.

- Descarga de pesos de entrenamiento: Los usuarios tienen la capacidad de descargar los pesos del entrenamiento seleccionado haciendo clic en el botón "Descargar Información de ID de Entrenamiento". Esto resulta útil para aquellos que deseen utilizar los pesos del modelo en análisis futuros o aplicaciones adicionales. Mientras se descarga la información se muestran por pantalla “Descargando…” y una vez ha terminado pone la frase de “Descarga finalizada con éxito”. Véase las Figuras 20 y 21.
- Volver al menú principal y restablecimiento del formulario: En cualquier momento, los usuarios pueden regresar al menú principal de la aplicación haciendo clic en el botón "Volver al Menú". Los usuarios también tienen la opción de comenzar de nuevo seleccionando otro modelo y entorno. Al hacer clic en el botón "Elegir otro modelo y entorno," se restablecen todos los estados, lo que permite a los usuarios iniciar un nuevo análisis desde cero. Los botones se encuentran en la parte inferior de la página web mientras que se encuentre el usuario en la parte de análisis de entrenamientos. Véase la Figura 16.
- Gestión de estados: El componente gestiona múltiples estados para rastrear la selección de modelo y entorno, el resumen de entrenamientos, la selección de ID de entrenamiento, la realización de *tests*, la descarga de pesos, entre otros. Esta gestión de estados asegura una experiencia de usuario coherente y receptiva.

En resumen, este componente proporciona una interfaz completa y rica en características para analizar y comprender los entrenamientos de modelos en diferentes entornos. Los usuarios pueden acceder a información detallada, gráficas y resultados de pruebas, lo que facilita la toma de decisiones informadas sobre sus modelos de aprendizaje por refuerzo.

Integración con backend a través de AXIOS

El propósito del *frontend* consiste en proporcionar una interfaz accesible al usuario, permitiéndole efectuar solicitudes y visualizar datos relevantes. Es importante destacar que existe una estrecha vinculación entre el *frontend* y el *backend*, y para gestionar eficazmente esta comunicación, se recurre a Axios, una biblioteca de JavaScript destinada a efectuar

solicitudes HTTP desde el cliente web hacia el servidor correspondiente. Axios asegura una comunicación fluida y segura, permitiendo que la aplicación web interactúe con el servidor para lograr sus objetivos funcionales. Esas solicitudes realizadas desde las componentes cargarDatos.js y analisisEntrenamientos.js tienen asociadas los *endpoints* mencionados en el capítulo de *backend* y que se resumen en la Figura 8.

Capítulo 6. Caso de uso

Descripción

Para mostrar el funcionamiento de la "Herramienta" en un ejemplo de aplicación de aprendizaje por refuerzo, se expone un caso de uso a través de los resultados y métricas obtenidos de los entrenamientos de un brazo robótico [22]. El entorno virtual incluye un brazo robótico IRB 120 de ABB con 6 grados de libertad, un alcance de 0,58 metros, una carga útil de 3 kg y una carga de brazo de 0,3 kg. En este caso, el objetivo del robot es alcanzar un objetivo cúbico rojo de 0,03 metros en un número máximo de pasos, que se considera un episodio.

El agente ha sido sometido a un proceso de entrenamiento en episodios. Al inicio de cada episodio, se han generado de manera aleatoria las posiciones del objetivo y del robot dentro de un espacio de trabajo definido. Cada episodio tiene una duración máxima de 50 pasos o finaliza si la distancia entre la pinza del robot y el objetivo es igual o menor a 5 centímetros.

En el proceso de entrenamiento, el agente ha acumulado experiencia a lo largo de un total de 70 millones de pasos. Cada vez que se alcanzan los 50 mil pasos, el entrenamiento se interrumpe para llevar a cabo 40 episodios de evaluación. Durante estos episodios de evaluación, se han calculado diversas métricas, incluyendo las distancias promedio, máxima y mínima desde la pinza del robot hasta la ubicación objetivo. Estos resultados se han utilizado para evaluar la efectividad del entrenamiento y estimar la eficiencia de la muestra, entre otras medidas.

Base de datos

Para el caso del diseño conceptual del caso de uso del robot, utilizaremos el diseño que se ha desarrollado en el capítulo 4. En ese esquema se encuentran las entidades y atributos clave necesarios para el desarrollo general de una aplicación de aprendizaje por refuerzo. Se añaden los atributos simples necesarios para guardar datos de los entrenamientos.

En la Tabla 6 se resumen las entidades y atributos para algoritmos de aprendizaje por refuerzo incluyendo la distinción por atributos de claves primarias, claves foráneas y atributos simples.

Tabla 6 Entidades y atributos para caso de uso con robot

Entidad	**Atributo**		
	Clave primaria	**Clave foránea**	**Atributo simple**
Model	id_model	-	description,date
Model Hyperparameters	id_modelH	id_model	type, name
Model Hyperparameters Str Value	id_modelHSV	id_model, id_modelH	value
Model Hyperparameters Txt Value	id_modelHTV	id_model, id_modelH	value
Environment	id_environment	-	description, date
Environment Characteristics	id_envCharacteristics	id_environment	connectionPartColor, shadow, robot, 1_5_robotColor, 6_robotColor, robotinitPose, groundColor, skyColor, fingersColor, camDistance, camElevation, camDegree, targetBoundsY, targetBoundsX, targetSize, targetShape, targetColor
Training	id_training	id_environment, id_model	seed, description, date
Training Hyperparameters	id_trainingH	id_training, id_model	type, name

Training Hyperparameters Str Value	id_trainingHSV	id_training, id_model, id_trainingH	value
Training Hyperparameters Num Value	id_trainingHNV	id_training, id_model, id_trainingH	value
Sampling	id_sampling	id_training	step_no
Gradients	id_gradients	id_sampling	blob
Weights	id_weights	id_sampling	blob
Test	id_test	id_model, id_weights, id_environment, id_test	stdAccuracy, stdEpLength, stdFailDist, maxFailDist, avgFailDist, avgAccuracy, avgEpLength
Episode	id_episode	id_test	tdError, epSteps, poseR, poseT
Step	id_test	id_step, id_episode	rewards, actions

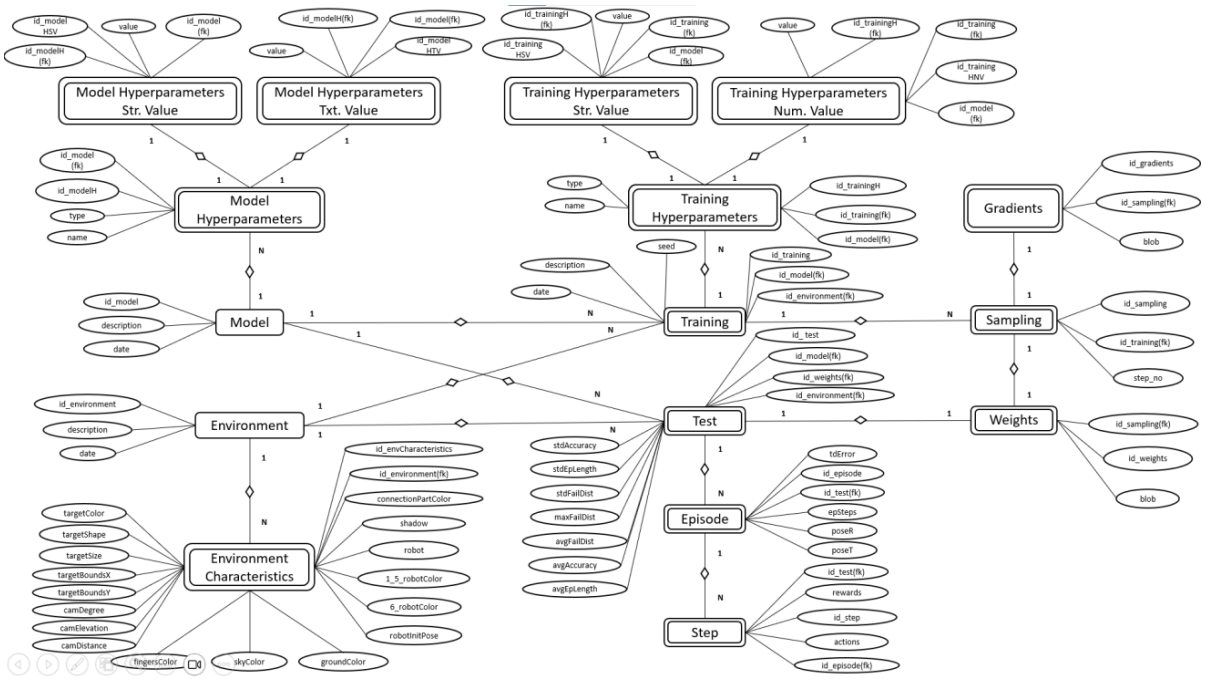


Figura 10 Diseño conceptual de un caso de uso con robot.

El diseño conceptual que se obtiene es el mostrado en la Figura 10.

Backend

Se implementa el diseño conceptual a un diagrama entidad-relación extendido que presenta MySQL Workbench para poder trasladar el ERM conceptual a un diagrama funcional. El resultado se muestra en la Figura 11.

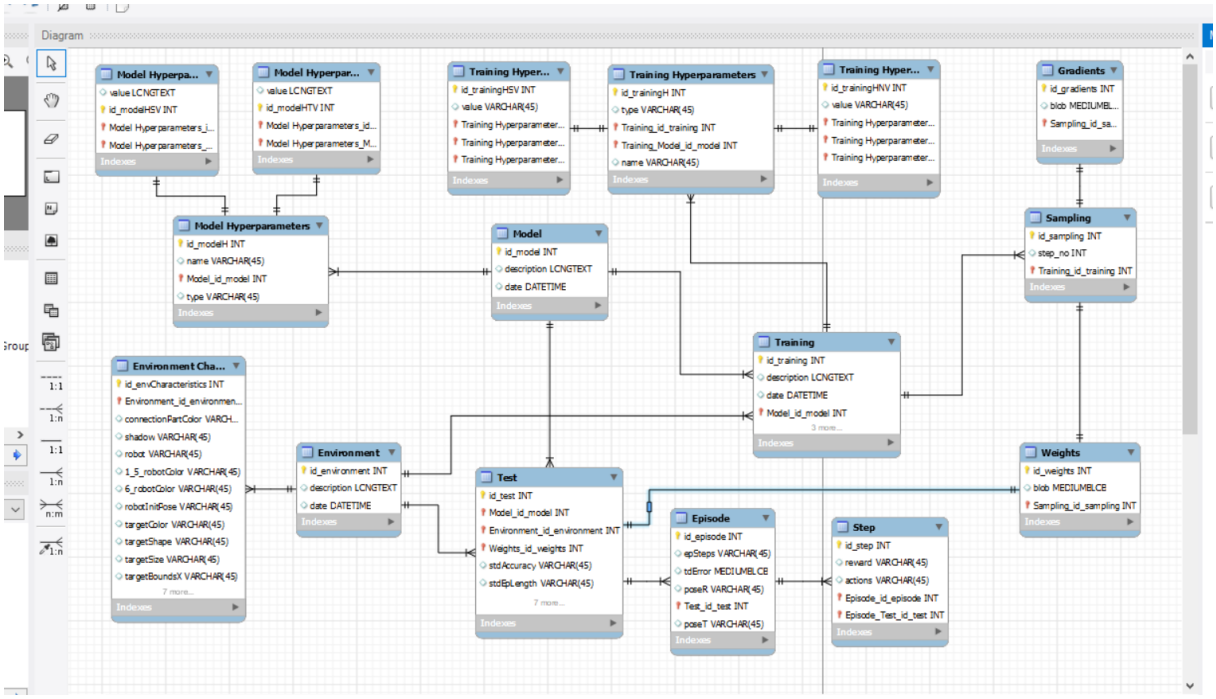


Figura 11 Diagrama EER de la BBDD para el caso de uso con robot.

El usuario carga la carpeta comprimida en zip con el contenido de los datos de entrenamientos a través de la interfaz de usuario. Una vez se realiza la carga se muestran datos ejemplos de las ingestas a la BBDD. Se han elegido la muestra de las entidades *Environment Characteristics* y *Model Hyperparameters Str. Value* (Figura 12).

El usuario desde el estado de carga de datos, en la página web, elige la carpeta comprimida con nombre “penvEasy_BDD” que ha generado el script de su código de entrenamiento. Posteriormente, se pulsa el botón cargar archivos y se muestra un mensaje al usuario, Figura 14, indicando que se está descargando y que dura aproximadamente 10 minutos la descarga, que es el tiempo medio de la carpeta de entrenamientos con un volumen de 5 GB.

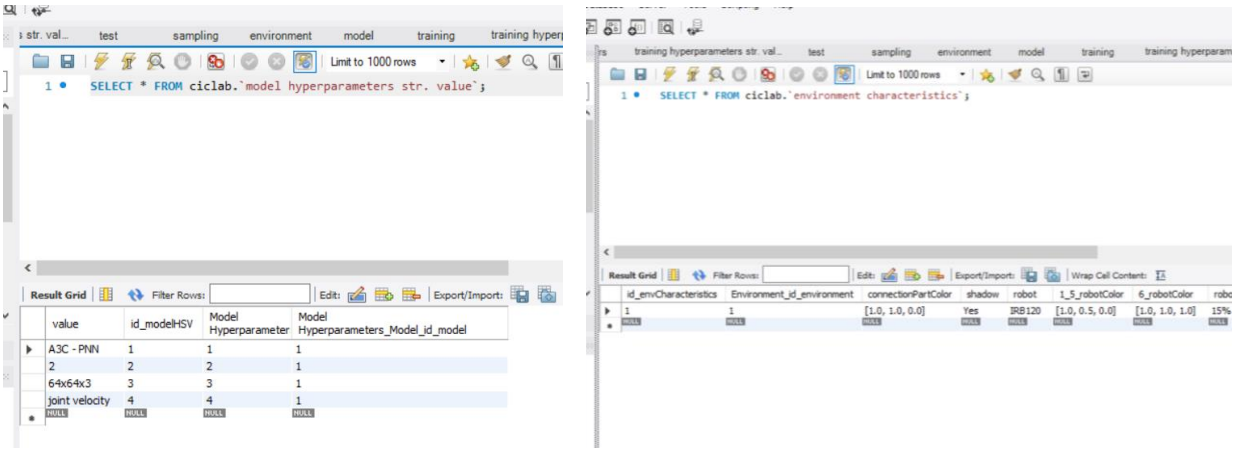


Figura 12 Ejemplo de muestra de valores de la BBDD del caso de uso con robot mediante queries de SQL para la tabla de la entidad Model Hyperparameters Str. Value en la izquierda y Environment Characteristics en la derecha.

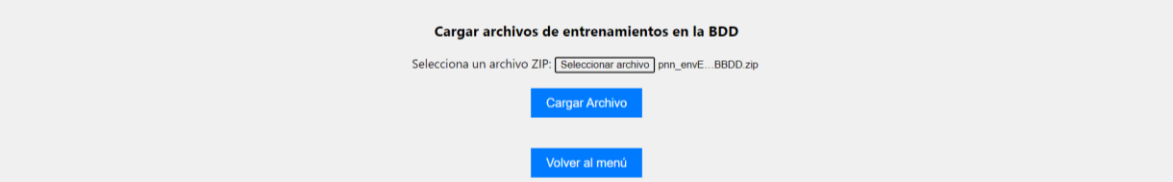


Figura 13 Interfaz del usuario en el estado de cargar archivos.

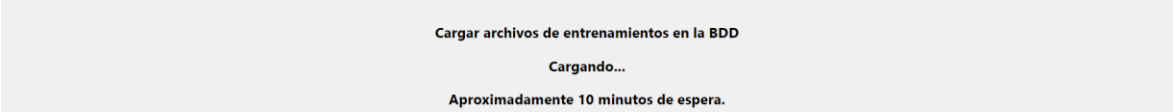


Figura 14 Interfaz del usuario cuando la carga de datos en la BBDD está en proceso.

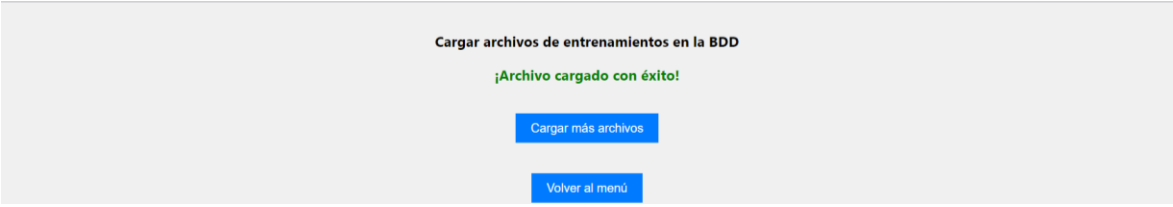
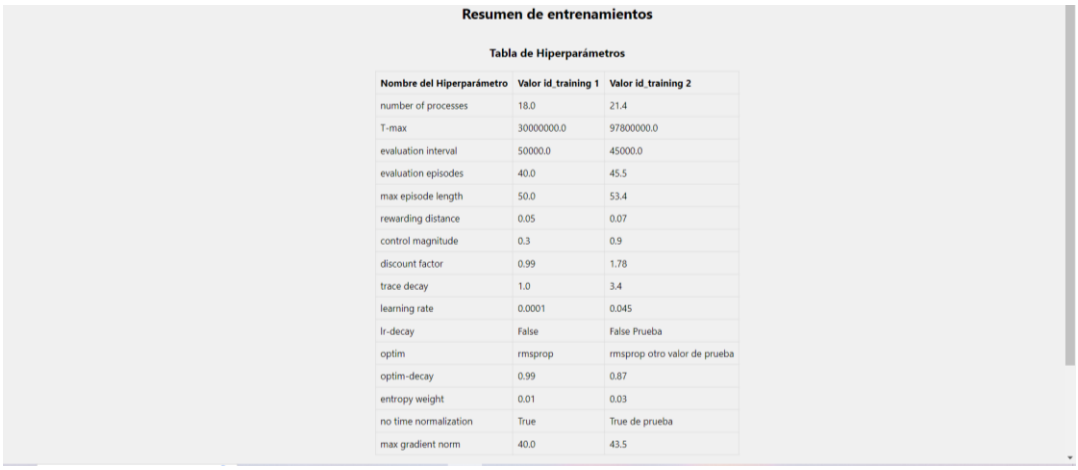


Figura 15 Interfaz del usuario cuando la carga de datos en la BBDD ha finalizado.

Figura 16 Interfaz del usuario en el estado de análisis de entrenamientos.

Frontend

El usuario tiene la opción de analizar los entrenamientos cargados en la BBDD. Se muestran los modelos y entornos cargados en la BBDD y en la Figura 17 aparece el resumen de entrenamientos asociados a una combinación de modelo y entorno. Según se observa en la Figura 18 el usuario elige uno de los dos entrenamientos, en este caso de ejemplo se elige el entrenamiento con ID 1, y puede ver las gráficas asociadas (Figura 19), puede ver los resultados de los *tests*, en el caso de la Figura 22 se elige el *test* con *step* número 50019, o descargar los pesos de los entrenamientos. El archivo descargado se encuentra en la carpeta del *backend* a nombre de "weights_entrenamiento_1.zip", por ser el número 1 el entrenamiento elegido por el usuario.



Nombre del Hiperparámetro	Valor id_training 1	Valor id_training 2
number of processes	18.0	21.4
T-max	30000000.0	97800000.0
evaluation interval	50000.0	45000.0
evaluation episodes	40.0	45.5
max episode length	50.0	53.4
rewarding distance	0.05	0.07
control magnitude	0.3	0.9
discount factor	0.99	1.78
trace decay	1.0	3.4
learning rate	0.0001	0.045
lr-decay	False	False Prueba
optim	rmsprop	rmsprop otro valor de prueba
optim-decay	0.99	0.87
entropy weight	0.01	0.03
no time normalization	True	True de prueba
max gradient norm	40.0	43.5

Figura 17 Interfaz del usuario mostrando los entrenamientos que coinciden con la combinación de modelo y entorno escogidas.

Figura 18 Interfaz de usuario en la elección del entrenamiento.

Figura 19 Interfaz de usuario mostrando las gráficas del entrenamiento seleccionado.

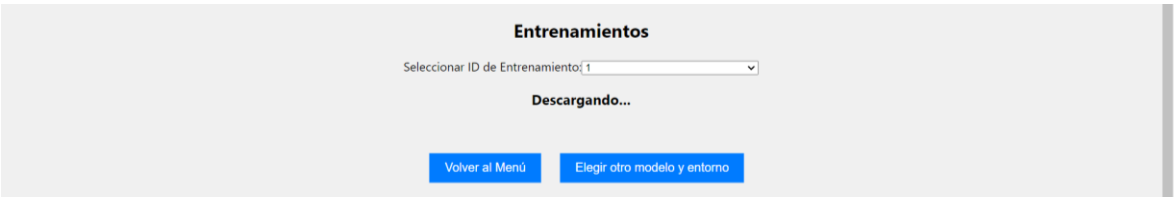


Figura 20 Interfaz de usuario en el proceso de descarga de los pesos del entrenamiento seleccionado.

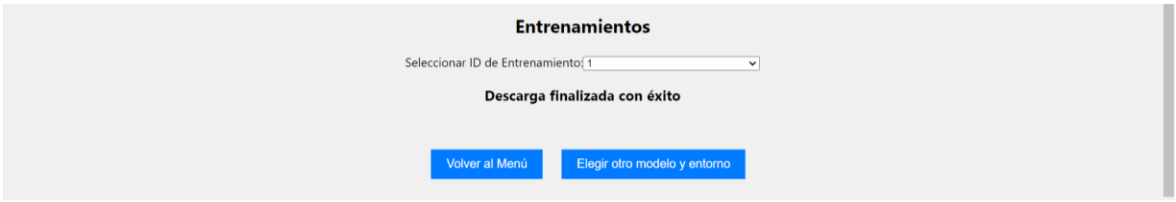


Figura 21 Interfaz de usuario con en el proceso de descarga de los pesos del entrenamiento seleccionado finalizado.

Figura 22 Interfaz de usuario mostrando los resultados del test seleccionado.

Capítulo 7. Conclusión y futuro desarrollo

La herramienta desarrollada es funcional y específica para el caso de uso mostrado en el proyecto, proporcionando soluciones precisas y adaptadas a ese contexto. Sin embargo, el método que emplea tiene un diseño robusto y versátil, lo que permite su generalización y aplicación a otros casos de aprendizaje por refuerzo.

Una de las características más destacadas de esta herramienta es su capacidad para cargar volúmenes de datos de 5 GB del entrenamiento del caso de uso en cuestión de aproximadamente 10 minutos. Además, la aplicación de esta herramienta se caracteriza por transiciones rápidas y fluidas. Esto asegura que el flujo de trabajo sea eficiente y sin interrupciones, lo cual es crucial para mantener la productividad y avanzar rápidamente en los proyectos. La combinación de estas características hace que la herramienta no solo sea útil para el caso específico para el que fue diseñada, sino también una opción viable y efectiva para una amplia variedad de situaciones en el ámbito del aprendizaje por refuerzo.

La herramienta puede seguir desarrollándose en mejoras como la incorporación de la BBDD a la nube unido de una mejoría del diseño de la interfaz de usuario en cuanto a visualización.

BBDD con servidor en la nube

La aplicación desarrollada permite trabajar desde local debido a la implementación de la BBDD en MySQL Workbench desde un servidor en el propio ordenador. Se había enfocado la utilidad de la aplicación para experimentos realizados por estudiantes donde toda la información se controla desde un ordenador. Esto limita la utilización de la aplicación a usuarios que no tengan acceso al servidor donde se ha desplegado la BBDD. La alternativa que se puede desarrollar, y que mejora la accesibilidad de la herramienta desarrollada, es poder tener la BBDD en la nube y de esta manera poder acceder desde cualquier punto del mundo. Esto presenta un mayor riesgo de seguridad al dar acceso a todos los datos de entrenamiento a una plataforma SaaS, cuya solución habitual sería usar VPN. Se debería llevar a cabo un estudio pertinente para ver si cumple con los requisitos marcados por el director del proyecto.

Mejora del diseño de la interfaz de usuario

El proyecto se ha centrado en asegurar la plena funcionalidad del sistema, asegurando que todas las características operativas cumplan con los estándares de eficiencia y eficacia requeridos. Sin embargo, un área identificada para futuras mejoras es la mejora de la interfaz de usuario. Optimizar el diseño y la presentación visual proporcionará una experiencia más estéticamente agradable, facilitando la interacción del usuario y mejorando la usabilidad general del sistema.

Bibliografía

[1] B Liu, B., Supervised Learning. In: "Web Data Mining. Data-Centric Systems and Applications", Springer, Berlin, Heidelberg (2011), Accedido Junio 2024; disponible *online* en: https://doi.org/10.1007/978-3-642-19460-3_3

[2] H.B. Barlow, "Unsupervised Learning. Neural Computation", 1989, Accedido Junio 2024; disponible *online* en: https://doi.org/10.1162/neco.1989.1.3.295

[3] Sutton, Richard S., and Andrew G. Barto, "Introduction to reinforcement learning", Vol. 135. Cambridge: MIT press, 1998.

[4] Weights & Biases, Weights & Biases Docs, Accedido Junio 2024; disponible *online* en: https://docs.wandb.ai/guides

[5] TensorFlow, TensorBoard, Accedido Junio 2024; disponible *online* en: https://www.tensorflow.org/learn?hl=es-419.

[6] Pytorch, Pytorch Documentation, Accedido Junio 2024; disponible *online* en: https://pytorch.org/docs/stable/index.html

[7] Keras, Keras Getting Started, Accedido Junio 2024; disponible *online* en: https://keras.io/getting_started/

[8] TensorFlow, TensorFlow Core, Accedido Junio 2024; disponible *online* en: https://www.tensorflow.org/guide?hl=es-419

[9] Angular, Angular Docs, Accedido Junio 2024; disponible *online* en: https://angular.io/docs

[10] Vue, Introduction to Vue, Accedido Junio 2024; disponible *online* en: https://vuejs.org/guide/introduction.html#what-is-vue

[11] React, React Docs, Accedido Junio 2024; disponible *online* en: https://legacy.reactjs.org/docs/getting-started.html

[12] Google, About Google, Accedido Junio 2024; disponible *online* en: https://about.google/

[13] Meta, Meta about us, Accedido Junio 2024; disponible *online* en: https://about.meta.com/company-info/

[14] 2022 State of JavaScript – Would use, Accedido Junio 2024; disponible *online* en: https://2022.stateofjs.com/en-US/libraries/front-end-frameworks/#front_end_frameworks_section_streams

[15] Angular vs React vs Vue: Which Framework to Choose, Accedido Junio 2024; disponible *online* en: https://www.codeinwp.com/blog/angular-vs-vue-vs-react/#gref

[16] Python, Python documentation, Accedido Junio 2024; disponible *online* en: https://www.python.org/doc/

[17] Django, Django documentation, Accedido Junio 2024; disponible *online* en: https://docs.djangoproject.com/en/4.2/

[18] FastAPI, FastAPI documentation, Accedido Junio 2024; disponible *online* en: https://fastapi.tiangolo.com/

[19] Flask, Flask user's guide, Accedido Junio 2024; disponible *online* en: https://flask.palletsprojects.com/en/2.3.x/#user-s-guide

[20] Entidad-Relación: Atributos. Tipos, Accedido Junio 2024; disponible *online* en: https://www.tuinstitutoonline.com/cursos/baseavanzado1_v1606/03atributos.php

[21] Plotly - Acerca de nosotros, Accedido Junio 2024; disponible *online* en: https://plotly.com/about-us/

[22] Güitta-López, L., Boal, J. & López-López, Á.J. Learning more with the same effort: how randomization improves the robustness of a robotic deep reinforcement learning agent. Appl Intell 53, 14903–14917 (2023). https://doi.org/10.1007/s10489-022-04227-3

Anexo ODS

Este proyecto se ajusta a los siguientes Objetivos de Desarrollo Sostenible:

- Objetivo 9 “Industria, innovación e infraestructura”: La herramienta desarrollada en este trabajo refleja una innovación generalizada para cualquier industria en el ámbito de visualización de algoritmos de aprendizaje por refuerzo. Definición objetivo: Construir infraestructura resiliente, promover la industrialización inclusiva y sostenible y fomentar la innovación.
- Objetivo 12 “Producción y consumo responsable”: La herramienta propuesta en el trabajo permite mediante la mejor monitorización de algoritmos una gestión eficiente en cuanto al uso de energía. Definición objetivo: Garantizar modalidades de consumo y producción sostenibles.
- Objetivo 13 “Cambio climático”: La herramienta desarrollada en el trabajo funciona a través de la red eléctrica y no consume directamente combustibles que puedan generar gases efecto invernadero al aire. Definición objetivo: Tomar medidas urgentes para combatir el cambio climático y sus impactos.