

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE INFORMÁTICA
SANTIAGO - CHILE



“COMPARACIÓN DE MODELOS DE LENGUAJE NATURAL PARA BÚSQUEDAS EN OBSERVATORIOS VIRTUALES”

JOSÉ FRANCISCO PINTO MUÑOZ

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL EN INFORMÁTICA

Profesor Guía: Mauricio Solar
Profesor Correferente: Mauricio Araya

Agosto - 2025



CONSTANCIA DE VALIDACIÓN Y CONFIDENCIALIDAD DE MONOGRAFÍA A REPOSITORIO ACADÉMICO

1.- IDENTIFICACIÓN DEL TRABAJO ACADÉMICO

Tipo de monografía (marcar una opción): ☒ Memoria o trabajo de título; ☐ Tesis de Postgrado;

Título del trabajo: Comparación de Modelos de Lenguaje Natural para Búsquedas en Observatorios Virtuales

Nombre del candidato(a): José Francisco Pinto Muñoz

Carrera / Grado: Ingeniería Civil Informática

Campus: Santiago San Joaquín; **Departamento:** Informática

2.- VALIDACIÓN DEL PROFESOR GUÍA/DIRECTOR DE TESIS

Yo, Mauricio Solar, en mi calidad de profesor(a) guía/director(a) del trabajo académico mencionado anteriormente **DEJO CONSTANCIA** que:

- He revisado esta versión del documento y corresponde a la versión final aprobada del trabajo.
- El trabajo cumple con los requisitos académicos y de formato establecidos por la institución

3.- EVALUACIÓN DE CONFIDENCIALIDAD POR PROPIEDAD INDUSTRIAL

☒ El trabajo **NO contiene información que amerite confidencialidad** y puede ser publicado de inmediato en repositorio con acceso abierto.

☐ El trabajo **CONTIENE** información con potenciales implicancias de propiedad industrial o intelectual y requiere un periodo de confidencialidad (embargo) por:

☐ 6 meses; ☐ 12 meses; ☐ 2 años; ☐ 3 años; ☐ 5 años; ☐ 10 años

Fundamentación de la necesidad de confidencialidad (obligatorio si se solicita embargo):

4.- FIRMAS

Profesor(a) guía o director(a) de memoria o tesis:

Fecha: 01/09/2025

; Firma:

Estudiante o Candidato(a):

Fecha: 01/09/2025

; Firma:

Este formulario debe ser insertado como página 2 de la memoria o tesis, completado y firmado por estudiante y profesor(a) antes de la entrega en portal PRISMA de Biblioteca USM.

DEDICATORIA

A Alan Turing, cuyo legado permitió que hoy las máquinas puedan ser inteligentes.

AGRADECIMIENTOS

Quisiera agradecer, en primer lugar, a mi familia, por estar siempre presente. A mi mamá, mi papá y mi hermana, por su cariño incondicional y por enseñarme el valor de la perseverancia. Gracias por permitirme dedicarme por completo a estudiar, apoyándome en todo momento y dándome la tranquilidad y confianza que no todos tienen la fortuna de recibir. Me lo dieron todo. También a mis abuelos, que sin saberlo, fueron parte de este camino al recordarlos.

A Sofía, por estar ahí en todas, con apoyo, sentido del humor y paciencia. Por motivarme, hacerme reír y recordarme que podía seguir, incluso en los días difíciles. Gracias por los cafecitos, por las pausas necesarias, y por acompañarme con cariño durante este proceso. Este trabajo también es un poco tuyo.

A Mila, mi perrita, que también fue parte del proceso, acompañándome durmiendo a mi lado mientras escribía y recordándome (mordiéndome) que a veces es necesario distraerse cuando empezaba a concentrarme demasiado.

A mis amigos, por ser empuje, compañía y distracción. Gracias por las risas, especialmente en momentos difíciles como la Feria de Software, que habrían sido imposibles de atravesar sin ustedes.

Agradezco también a quienes me ayudaron a pensar esta memoria. A mi profesor guía, por confiar en esta propuesta y darme la libertad para pensar con creatividad. Y, por último, a quienes abrieron camino antes que yo: gracias por dejar memorias que también me guiaron cuando no sabía por dónde seguir.

RESUMEN

Resumen— Este trabajo presenta el desarrollo y evaluación de VOAI-Search, una plataforma para consultas en lenguaje natural sobre Observatorios Virtuales astronómicos. Se implementó una arquitectura con frontend interactivo, backend de coordinación, base de datos, motor de búsqueda compatible con protocolos SIAP de IVOA y dos modelos LLM: GPT-4o y DeepSeek-R1. El objetivo fue comparar su desempeño en recuperación de imágenes astronómicas, evaluando precisión, tiempo de respuesta, recursos y sostenibilidad. Se realizaron 21 consultas por modelo agrupadas temáticamente, midiendo métricas como Recall@1, Recall@3, MRR, tokens, costo y huella de carbono. DeepSeek-R1 logró mayor Recall@3 (86 % vs. 71 %), menor costo e impacto ambiental, mientras GPT-4o fue más rápido y usó menos tokens. La evaluación demuestra fortalezas diferenciadas que mejoran el acceso semántico a datos astronómicos.

Palabras Clave— Observatorio Virtual; Modelos de Lenguaje; Astroinformática; GPT-4o; DeepSeek

ABSTRACT

Abstract— This work presents the development and evaluation of VOAI-Search, a web platform enabling natural language queries on astronomical Virtual Observatories using state-of-the-art LLMs. The architecture includes an interactive frontend, coordinating backend, database, search engine compatible with IVOA's SIAP protocols, and two LLMs: GPT-4o and DeepSeek-R1. The main goal was to compare their performance in retrieving astronomical images, assessing precision, response time, resource consumption, and sustainability. Twenty-one thematic queries per model were executed, measuring Recall@1, Recall@3, MRR, tokens, cost, and carbon footprint. DeepSeek-R1 achieved higher Recall@3 (86 % vs. 71 %), lower cost and environmental impact, while GPT-4o was faster and used fewer tokens. The evaluation shows distinct strengths enhancing semantic access to large astronomical datasets.

Keywords— Virtual Observatory; Language Models; Astroinformatics; GPT-4o; DeepSeek

GLOSARIO

ALMA: Atacama Large Millimeter/submillimeter Array.
ANN: Artificial Neural Network.
API: Application Programming Interface.
API REST: Representational State Transfer Application Programming Interface.
AURA: Association of Universities for Research in Astronomy.
CNN: Convolutional Neural Network.
CTIO: Cerro Tololo Inter-American Observatory.
Dec: Declination.
DECam: Dark Energy Camera.
DOI: Digital Object Identifier.
DPO: Direct Preference Optimization.
ESO: European Southern Observatory.
FITS: Flexible Image Transport System.
GPT: Generative Pre-trained Transformer.
HTTP: Hypertext Transfer Protocol.
IA: Inteligencia Artificial.
IVOA: International Virtual Observatory Alliance.
JPEG: Joint Photographic Experts Group.
JSON: JavaScript Object Notation.
LLM: Large Language Models.
ML: Machine Learning.
MLA: Multi-head Latent Attention.
MoE: Mixture of Experts.
MRR: Mean Reciprocal Rank.
NLP: Natural Language Processing.
NOIRLab: National Optical-Infrared Astronomy Research Laboratory.
ObsCore: Observation Core Data Model.
PNG: Portable Network Graphics.
RA: Right Ascension.
ReLU: Rectified Linear Unit.
RNN: Recurrent Neural Network.
ROUGE: Recall-Oriented Understudy for Gisting Evaluation.
SCS: Simple Cone Search.
SED: Spectral Energy Distribution.
SFT: Supervised Fine Tuning.
SIAP: Simple Image Access Protocol.
SSA: Service Spectral Protocol.
TAP: Table Access Protocol.
WWW: World Wide Web.
XML: Extensible Markup Language.

ÍNDICE DE CONTENIDOS

RESUMEN	IV
ABSTRACT	IV
GLOSARIO	V
ÍNDICE DE FIGURAS	VIII
ÍNDICE DE CÓDIGOS	VIII
ÍNDICE DE TABLAS	IX
1 INTRODUCCIÓN	1
2 DEFINICIÓN DEL PROBLEMA	3
2.1 BIG DATA EN LA ASTRONOMÍA	3
2.2 IVOA Y OBSERVATORIOS VIRTUALES	4
2.3 OBJETIVOS	5
2.3.1 OBJETIVO GENERAL	5
2.3.2 OBJETIVOS ESPECÍFICOS	5
2.4 ALCANCE	6
3 MARCO CONCEPTUAL	7
3.1 DEFINICIÓN DE OBSERVATORIO VIRTUAL	7
3.2 BÚSQUEDA DE LA INFORMACIÓN	9
3.2.1 SUBSECCIÓN "REGISTRY"	9
3.2.2 SUBSECCIÓN "DATA ACCESS PROTOCOL STANDARDS"	10
3.2.3 PROTOCOLO SIA Y FORMATO DE IMAGENES	11
3.2.4 PYVO Y PETICIONES SIAP	13
3.3 DESDE IA A PROCESAMIENTO DE LENGUAJE NATURAL	15
3.3.1 INTRODUCCIÓN A LA INTELIGENCIA ARTIFICIAL (IA)	15
3.3.2 APRENDIZAJE AUTOMÁTICO	16
3.3.3 REDES NEURONALES ARTIFICIALES	17
3.3.4 PROCESAMIENTO DE LENGUAJE NATURAL	18
3.4 CHATGPT y DEEPSEEK	19
3.4.1 OPENAI: GPT Y O1	19
3.4.2 DEEPSEEK-V2 Y DEEPSEEK-R1	21
3.5 CHATBOTS CON LLM	23
3.5.1 PROMPTS	23
3.5.2 EVALUACIONES	24
3.5.3 IMPACTO SOCIAL Y AMBIENTAL	25

3.6	HERRAMIENTAS DE DESARROLLO WEB	26
3.6.1	STACK PARA DESARROLLO DE PLATAFORMAS	26
3.7	AGENTE ASTRONÓMICO	27
4	PROPUESTA DE SOLUCIÓN	28
4.1	ARQUITECTURA GENERAL	28
4.1.1	DESCRIPCIÓN DEL STACK TECNOLÓGICO	30
4.2	DISEÑO Y DESARROLLO DE FRONTEND	31
4.2.1	IDENTIDAD VISUAL	31
4.2.2	INTERFAZ	33
4.3	DISEÑO Y DESARROLLO DE BACKEND	36
4.3.1	BASE Y MODELO DE DATOS	36
4.3.2	RUTAS Y LÓGICA DE API	37
4.4	INTEGRACIÓN CON MODELOS LLM	38
4.4.1	LÓGICA DEL MOTOR DE BÚSQUEDA	39
4.5	FLUJO DE DATOS	41
4.6	MÉTRICAS DE EVALUACIÓN	42
4.6.1	RECALL@K	42
4.6.2	MRR	43
4.6.3	TIEMPO DE RESPUESTA	43
4.6.4	COSTO POR CONSULTA	43
4.6.5	HUELLA DE CARBONO ESTIMADA POR CONSULTA	44
4.6.6	ROUGE	45
4.7	PLANIFICACIÓN DE PRUEBAS Y PROCESAMIENTO DE RESULTADOS	45
4.7.1	CONSULTAS	46
4.7.2	EJECUCIÓN DE CONSULTAS	48
4.7.3	DIMENSIONES A EVALUAR	48
5	RESULTADOS Y ANÁLISIS	50
5.1	RESULTADOS DE LAS PRUEBAS	50
5.1.1	ENTORNO Y METODOLOGÍA DE EJECUCIÓN	50
5.1.2	RESULTADOS CONSOLIDADOS POR MODELOS	51
5.1.3	COMPARACIÓN ENTRE MODELOS	52
5.1.4	ANÁLISIS COMPARATIVO	53
6	CONCLUSIONES	55
6.1	CUMPLIMIENTO DE OBJETIVOS	55
6.2	APLICACIONES Y RECOMENDACIONES	56
6.3	TRABAJO FUTURO	56
	ANEXOS	57
	ANEXOS	57
	REFERENCIAS BIBLIOGRÁFICAS	64

ÍNDICE DE FIGURAS

3.1 Nivel 0 Arquitectura de IVOA.	8
3.2 Nivel 1 Arquitectura de IVOA.	9
3.3 Estructura de una neurona artificial	17
3.4 Arquitectura del Transformer.	19
3.5 Evaluación MT-Bench DeepSeek LLM.	22
4.1 Diagrama de la Arquitectura de VOAI-Search.	29
4.2 Logo VOAI Search.	32
4.3 Interfaz de escritorio.	34
4.4 Interfaz móvil.	35

ÍNDICE DE CÓDIGOS

3.1	Petición SIAP en PyVO a Registro específico.	13
3.2	Búsqueda en el registro de los servicios que implementan SIAP	15

ÍNDICE DE TABLAS

5.1	Resultados promedio por ítem temático - GPT-4o	51
5.2	Resultados promedio por ítem temático - DeepSeek-R1	52
5.3	Comparación general de métricas entre GPT-4o y DeepSeek-R1	52
6.1	Resultados individuales por consulta para el Ítem Temático <i>Galaxia NGC 1300</i> - GPT-4o	57
6.2	Resultados individuales por consulta para el Ítem Temático <i>Nebulosa del Cangrejo</i> - GPT-4o	57
6.3	Resultados individuales por consulta para el Ítem Temático <i>Región celeste específica</i> - GPT-4o	58
6.4	Resultados individuales por consulta para el Ítem Temático <i>Centro galáctico en infrarrojo</i> - GPT-4o	58
6.5	Resultados individuales por consulta para el Ítem Temático <i>Colisiones y fenómenos físicos</i> - GPT-4o	59
6.6	Resultados individuales por consulta para el Ítem Temático <i>Cúmulos y formación estelar</i> - GPT-4o	59
6.7	Resultados individuales por consulta para el Ítem Temático <i>Emisiones específicas y eventos astronómicos</i> - GPT-4o	60
6.8	Resultados individuales por consulta para el Ítem Temático <i>Galaxia NGC 1300</i> - DeepSeek-R1	60
6.9	Resultados individuales por consulta para el Ítem Temático <i>Nebulosa del Cangrejo</i> - DeepSeek-R1	61

6.10	Resultados individuales por consulta para el <i>Ítem Temático Región celeste específica</i> – DeepSeek-R1	61
6.11	Resultados individuales por consulta para el <i>Ítem Temático Centro galáctico en infrarrojo</i> – DeepSeek-R1	62
6.12	Resultados individuales por consulta para el <i>Ítem Temático Colisiones y fenómenos físicos</i> – DeepSeek-R1	62
6.13	Resultados individuales por consulta para el <i>Ítem Temático Cúmulos y formación estelar</i> – DeepSeek-R1	63
6.14	Resultados individuales por consulta para el <i>Ítem Temático Emisiones específicas y eventos astronómicos</i> – DeepSeek-R1	63

Capítulo 1

INTRODUCCIÓN

La astronomía es la ciencia que estudia los cuerpos celestes y los fenómenos que ocurren fuera de la atmósfera terrestre. A lo largo del tiempo, esta disciplina ha evolucionado gracias al desarrollo tecnológico, integrando herramientas digitales que permiten observar, analizar y entender el universo de forma más profunda. En particular, el vertiginoso crecimiento en la generación de datos provenientes de observatorios y simulaciones ha derivado en una nueva etapa en la ciencia astronómica.

La astroinformática surge como una disciplina interdisciplinaria que combina astronomía, ciencia de datos, informática, y tecnologías de la información, con el objetivo de gestionar y analizar volúmenes masivos de información. Esta área involucra el desarrollo de software especializado, redes neuronales, sistemas distribuidos y plataformas interactivas, todos ellos enfocados en facilitar la labor investigativa. En este escenario, los Observatorios Virtuales representan una pieza clave: permiten el acceso remoto a bases de datos astronómicos mediante estándares abiertos definidos por la International Virtual Observatory Alliance, haciendo posible la interoperabilidad entre múltiples centros de datos.

No obstante, el acceso a estos recursos sigue presentando desafíos importantes. Si bien los VO permiten acceder a grandes volúmenes de información astronómica, su uso requiere conocimientos técnicos avanzados sobre los protocolos y estructuras utilizadas. Esta situación genera una barrera para usuarios no especializados, dificultando la exploración y análisis eficiente de los datos disponibles. Frente a esta problemática, surge la necesidad de interfaces más accesibles que permitan interactuar con los Observatorios Virtuales utilizando lenguaje natural.

Esta memoria aborda dicha necesidad mediante el diseño e implementación de VOAI-Search, una plataforma que integra tecnologías de inteligencia artificial, específicamente modelos de lenguaje de última generación (LLMs), para facilitar el acceso semántico a datos astronómicos. Se emplean modelos como GPT-4o y DeepSeek-R1, permitiendo la formulación de consultas en lenguaje natural y la recuperación de información relevante desde repositorios

compatibles con IVOA, en específico imágenes mediante el protocolo SIA. Esta solución busca transformar la experiencia de búsqueda, permitiendo una interacción intuitiva entre el usuario y los datos.

La propuesta considera el desarrollo de una aplicación web compuesta por un frontend amigable, un backend que orquesta las consultas y un motor de búsqueda basado en modelos LLM. Además, se evalúa el desempeño de los modelos utilizados bajo métricas como Recall, MRR, tiempo de respuesta, tokens consumidos, costo por consulta y huella de carbono estimada, aplicadas a un conjunto de 21 consultas por modelo agrupadas en siete categorías astronómicas.

Este documento se organiza de la siguiente manera: el Capítulo 2 presenta el contexto del problema, sus objetivos y el alcance del trabajo. El Capítulo 3 introduce los conceptos clave relacionados con los observatorios virtuales, inteligencia artificial, modelos de lenguaje y herramientas de desarrollo utilizadas. El Capítulo 4 detalla la arquitectura y diseño del sistema propuesto. El Capítulo 5 expone la validación de la solución mediante pruebas y análisis comparativo entre modelos. Finalmente, el Capítulo 6 incluye las conclusiones, recomendaciones y proyecciones para trabajos futuros.

Capítulo 2

DEFINICIÓN DEL PROBLEMA

Chile es conocido mundialmente por tener uno de los mejores cielos para la astronomía, especialmente para la investigación astronómica. Esto se debe a las altas cumbres y planicies, las cuales tienen una baja contaminación lumínica, cielos despejados y baja humedad, en particular en las regiones del Norte Grande y Norte Chico. Estas condiciones naturales y al trabajo conjunto entre el estado, privados, casas de estudio, organizaciones de investigación nacionales e internacionales, ha permitido que se establezcan importantes **observatorios astronómicos** en el país, posicionando a Chile como una de las principales capitales de la astronomía a nivel global. Algunos de los observatorios en Chile ¹ incluyen:

ALMA “Atacama Large Millimeter/submillimeter Array”, una colaboración internacional que cuenta con 66 antenas para observar fenómenos astronómicos a cientos de años luz. El **Observatorio Paranal**, gestionado por el “European Southern Observatory” (ESO), que opera varios telescopios ópticos combinados que han sido fundamentales en el estudio de agujeros negros y planetas extrasolares. El **Observatorio Las Campanas**, parte de los Observatorios Carnegie, que alberga los telescopios gemelos Magallanes y ha sido clave en la astronomía moderna, especialmente en proyectos como el “Sloan Digital Sky Survey”. El **Observatorio Interamericano de Cerro Tololo** (CTIO), administrado por la Asociación de Universidades para la Investigación en Astronomía (AURA), que ha sido un pilar de la investigación astronómica en el hemisferio sur, destacando con la Cámara de Energía Oscura (DECam). *Gemini Sur*, parte del **Observatorio Internacional Gemini**, es uno de los telescopios más avanzados del mundo en óptica infrarroja, localizado en la Región de Coquimbo.

2.1 BIG DATA EN LA ASTRONOMÍA

Una de las condiciones que se establecen para los observatorios en Chile es que deben reservar al menos “el 10 % del tiempo de observación de telescopios a proyectos chilenos. Se le

¹[Observatorios de Chile: los ojos del mundo al Universo](#)

*concede status diplomático y exención de impuestos*². Con esto, estas instalaciones aportan al progreso astronómico de la comunidad científica.

Es por esto que los observatorios astronómicos enfrentan un desafío creciente debido al gran volumen de datos generados por sus sofisticados instrumentos. Proyectos como ALMA, producen volúmenes de datos que llegan a escalar hasta petabytes por año, lo que requiere una gestión avanzada que va más allá de los sistemas tradicionales. La capacidad de captura y análisis de estos datos no sólo es crítica para el avance científico, sino también para manejar la variedad, velocidad y veracidad de la información que se produce. Este fenómeno, denominado “**Big Data**”, obliga a los observatorios a adoptar tecnologías de computación en la nube y procesamiento distribuido, con el fin de procesar y analizar los datos de manera eficiente. El enfoque del “Big Data” se basa en el manejo de **las “5 V”: volumen, velocidad, veracidad, variedad y valor** [Patgiri y Ahmed, 2016].

El concepto de “Big Data”, más que referirse a una tecnología específica, implica un enfoque o metodología para extraer valor de los grandes volúmenes de información que se generan en la actualidad. Se deben considerar aspectos esenciales como la captura, gestión y explotación de los datos, asegurando su validez y fiabilidad, así como el compartir y comunicar la información para mejorar la toma de decisiones y facilitar el análisis [Solar et al., 2015]. En particular, un área donde el desafío del “Big Data” ha alcanzado un punto crítico es la astronomía.

2.2 IVOA Y OBSERVATORIOS VIRTUALES

Los **Observatorios Virtuales** (VO, por sus siglas en inglés) surgieron como una solución necesaria ante el creciente problema del manejo y almacenamiento de grandes volúmenes de datos en astronomía, fenómeno conocido como “Big Data”. Se puede decir que un VO en astronomía es un concepto para almacenar los datos astronómicos en archivos digitales y proporcionar acceso internacional a ellos [Vavilova et al., 2016].

Para hacer frente a este problema del “Big Data” en la astronomía, se creó la **International Virtual Observatory Alliance (IVOA)** en el 2002, una organización que busca estandarizar y unificar los datos procedentes de los distintos observatorios alrededor del mundo. Su objetivo principal es desarrollar una plataforma de colaboración global donde los datos astronómicos puedan ser fácilmente accesibles y procesados, sin importar el lugar en que se encuentren almacenados. Esta iniciativa buscaba estandarizar la interoperabilidad entre los diferentes observatorios astronómicos alrededor del mundo, permitiendo que los datos recogidos puedan ser utilizados de manera eficiente por investigadores, independientemente de su ubicación o del sistema de almacenamiento utilizado por cada observatorio [Solar et al., 2015]. Uno de los criterios que establece IVOA para pertenecer al VO, es poder agregar datos y crear aplicaciones que accedan a estos datos.

²CONICYT, Observatorios astronómicos en Chile, Santiago, 2008

Cada VO cuenta con su propio sistema de búsqueda interno, compuesto por uno o más subsistemas que permiten el acceso a los datos administrados por el observatorio [Araya et al., 2015]. Un VO es, en esencia, un conjunto de servicios de datos que operan bajo reglas comunes, lo que facilita la interoperabilidad entre diferentes observatorios. La principal ventaja de estos servicios estandarizados es que los usuarios no necesitan acceder a numerosos sitios web ni aprender diversas interfaces para consultar los datos. En su lugar, pueden utilizar su herramienta o portal favorito, lo que permite obtener acceso a cientos de conjuntos de datos de manera unificada. Este enfoque estandarizado asegura que las cajas de parámetros de búsqueda sean similares entre diferentes servicios, lo que simplifica la experiencia del usuario.

Sin embargo, este sistema también presenta oportunidades. Aunque el objetivo del IVOA es facilitar el acceso a los datos, cada servicio de datos puede tener su propia semántica, lo que obliga a los usuarios a adaptarse a las especificidades de cada uno. Además, la mayoría de estos sistemas no permite la búsqueda mediante lenguaje natural, lo que añade una barrera adicional para los usuarios que no están familiarizados con las técnicas avanzadas de consulta. Este desajuste entre la estandarización y la diversidad en la semántica de búsqueda podría convertirse en un obstáculo para cumplir con los objetivos de accesibilidad planteados por el IVOA.

2.3 OBJETIVOS

2.3.1. OBJETIVO GENERAL

El objetivo general de este trabajo consiste en comparar el desempeño de dos modelos de inteligencia artificial generativa, *GPT-4o* y *DeepSeek-R1*, en la implementación de un buscador optimizado para consultas en Observatorios Virtuales, evaluando su eficacia en términos de rapidez y precisión en el contexto de búsqueda científica, especialmente en imágenes astronómicas.

2.3.2. OBJETIVOS ESPECÍFICOS

Para poder lograr el objetivo general se hace un desglose de subobjetivos más específicos que ayudarán a validar el cumplimiento del trabajo. Estos objetivos son:

- Comprender las características y diferencias estructurales entre los modelos *GPT-4o* y *DeepSeek-R1* para determinar los factores que impactan en la velocidad, calidad y costo de respuesta y en la generación de respuestas contextualizadas en búsquedas científicas.

- Diseñar e implementar una interfaz que integre ambos modelos, desarrollando un entorno donde se puedan realizar pruebas de consulta en tiempo real y comparar los resultados entre el modelo de *OpenAI* y el *DeepSeek*.
- Analizar en diferentes términos las respuestas del de la interfaz, evaluando la idoneidad de cada modelo para las necesidades específicas de los usuarios de los Observatorios Virtuales.

2.4 ALCANCE

Este trabajo se enfoca en el desarrollo y evaluación de una plataforma que utiliza un modelo de inteligencia artificial generativa orientado a facilitar la búsqueda y consulta de imágenes en los Observatorios Virtuales mediante el uso de lenguaje natural. El documento está dirigido tanto a investigadores y astrónomos que deseen realizar consultas eficientes en los Observatorios Virtuales, como a ingenieros informáticos interesados en explorar soluciones basadas en inteligencia artificial para la interacción con grandes volúmenes de datos astronómicos. Asimismo, ofrece una base sobre el uso de modelos generativos para el uso de lenguaje natural en consultas especializadas en el contexto de la astronomía.

Capítulo 3

MARCO CONCEPTUAL

El presente capítulo establece los fundamentos teóricos y conceptuales sobre los cuales se basa la solución propuesta en esta memoria. A través de la definición de conceptos clave y su interrelación con la problemática abordada, se busca delimitar el contexto de la investigación.

3.1 DEFINICIÓN DE OBSERVATORIO VIRTUAL

Un VO tiene como objetivo proporcionar a los astrónomos un acceso transparente y sin interrupciones a múltiples centros de datos astronómicos, de manera similar a como la “World Wide Web”(WWW) permite a los usuarios acceder a documentos de todo el mundo a través de enlaces. Al igual que la web, el VO no es un sistema fijo, sino una forma de hacer las cosas .

El **concepto de VO** se puede entender como un ecosistema de conjuntos de datos, servicios y herramientas de software interconectados a través de un conjunto común de tecnologías y estándares. No es un sistema físico como un almacén de datos, sino una red interoperable que permite a los astrónomos acceder y analizar información de múltiples centros de datos. El sistema de todo VO debe adaptarse a los **estándares promovidos por la IVOA** con el fin de conseguir interoperabilidad en los servicios y herramientas, por lo tanto, es importante conocer los protocolos de acuerdo a los objetivos que persigue las plataformas astronómicas.

La **arquitectura de un VO** está diseñada para facilitar el acceso, la interoperabilidad y la explotación de datos astronómicos distribuidos en diferentes bases de datos y servicios, y se divide en los siguientes: El **Nivel 0** ofrece una vista general del ecosistema del Observatorio Virtual y puede entenderse como una estructura dividida en tres capas (Ver figura  3.1).

¹What is the VO?

- **Capa de Recursos:** Es la responsable de almacenar los datos procedentes de diversas fuentes, como observatorios, telescopios y simulaciones teóricas. La información almacenada incluye imágenes astronómicas, catálogos, espectros y otros tipos de datos. Esta capa también puede incluir sistemas de almacenamiento distribuidos en la nube para gestionar grandes volúmenes de datos.
- **Capa de Usuarios:** Esta capa representa la interfaz a través de la cual los investigadores y usuarios interactúan con el sistema. Aquí se encuentran las aplicaciones y herramientas que permiten consultar, visualizar y analizar los datos astronómicos.
- **Capa Intermedia:** Actúa como el nexo que conecta las capas de usuarios y recursos de manera transparente. Esta capa incluye protocolos y estándares que aseguran una interacción fluida entre los usuarios y los recursos. Aquí, el VO gestiona la búsqueda de recursos, el acceso a los datos y la integración de servicios distribuidos.

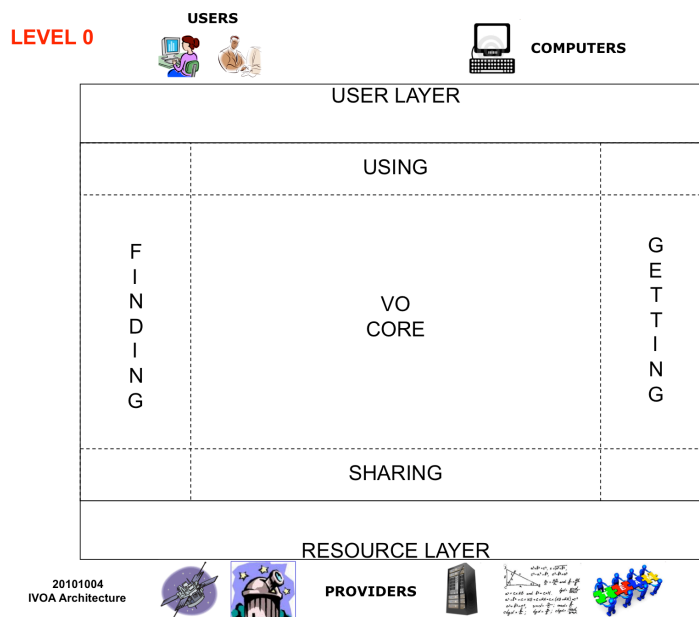


Figura 3.1: Nivel 0 Arquitectura de IVOA.

Fuente: <https://www.ivoa.net/deployers/>

Mientras que el **Nivel 1**, se profundiza en las funciones específicas que realiza cada capa del sistema [3.2](#)).

- **Encontrar:** Se encarga de aspectos esenciales para la gestión de registros dentro del sistema y cómo se presentan en las Páginas Amarillas de la VO. Esto incluye la normativa para nombrar recursos de manera coherente y cómo listar los metadatos asociados a estos recursos. Además, se especifica cómo los usuarios pueden interactuar con un registro, proporcionando las directrices para su manipulación y consulta.

- **Obtener:** Esta capa tiene un enfoque en los protocolos de acceso a datos. Define los métodos y estándares necesarios para la recuperación de información dentro del sistema, asegurando que el acceso a los datos se realice de manera uniforme y eficiente. Estos protocolos son cruciales para mantener la integridad y la disponibilidad de la información.
- **Núcleo de la VO:** Se centra en especificar los diferentes tipos de modelos de datos permitidos, la semántica que debe ser seguida, y cómo construir consultas de manera estandarizada. Esta capa proporciona las bases para la interpretación de datos y la creación de consultas, estableciendo un marco que facilita la interoperabilidad y la coherencia entre los diferentes componentes del sistema. Estos detalles están dirigidos principalmente a proveedores de servicios y desarrolladores de aplicaciones, quienes necesitan estos lineamientos para construir y mantener el sistema de manera efectiva.

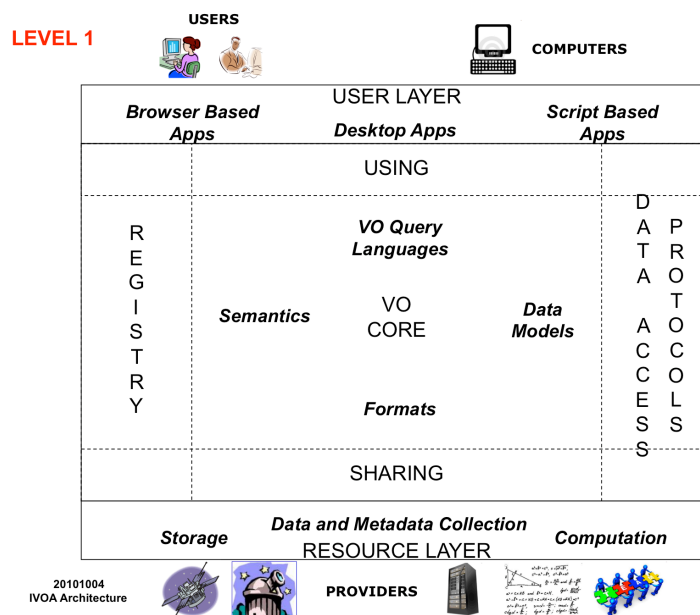


Figura 3.2: Nivel 1 Arquitectura de IVOA.

Fuente: <https://www.ivoa.net/deployers/>

3.2 BÚSQUEDA DE LA INFORMACIÓN

3.2.1. SUBSECCIÓN “REGISTRY”

En la capa intermedia del Observatorio Virtual, la subsección “REGISTRY” es fundamental para facilitar el acceso a colecciones de datos, metadatos y servicios distribuidos. Un recurso VO abarca una amplia gama de elementos, desde conceptos abstractos hasta imágenes y

catálogos, que pueden ser relevantes para los astrónomos. Estos recursos se describen con un nombre, un identificador único y un responsable de su mantenimiento.

El “VOResource” es un estándar **XML** que se utiliza para describir estos recursos. Ofrece un formato de intercambio XML para facilitar la transferencia de descripciones entre elementos VO, editores y registros. Este estándar permite una comunicación eficiente y estandarizada sobre los recursos disponibles [Molina, 2017].

“Registry Interface”, por su parte, define las interfaces para la comunicación e interacción de los registros con las aplicaciones. Este modelo distribuido incluye dos tipos de registros: **Registros de búsqueda y Registros de publicación**. Los registros de búsqueda permiten a los usuarios localizar recursos mediante la búsqueda de metadatos en diversos registros. Por otro lado, los registros de publicación estandarizan la oferta de recursos para que puedan ser descubiertos por registros de búsqueda.

Además, “Resource Metadata” se refiere a la descripción detallada de un recurso VO, similar a una “página amarilla” para la información astronómica. Este estándar promueve la independencia de los elementos VO mediante descripciones exhaustivas.

3.2.2. SUBSECCIÓN “DATA ACCESS PROTOCOL STANDARDS”

En la subsección “DATA ACCESS PROTOCOL STANDARDS” se centra en proporcionar servicios de consulta y acceso a colecciones distribuidas de metadatos y datos. Esta interfaz permite a los agentes de la capa de usuario acceder a datos almacenados en diversas ubicaciones. Cada estándar de protocolo está diseñado para manejar tipos específicos de datos, como imágenes astronómicas o espectros. Alguno de estos agentes son:

- **SIAP**(“Simple Image Access Protocol”): Este protocolo está diseñado para facilitar la búsqueda y recuperación de imágenes astronómicas. Las imágenes pueden variar en tamaño y complejidad, desde imágenes bidimensionales hasta cubos de datos espectrales e hipercubos. Utilizando SIAP, se puede definir una región rectangular en el cielo para consultar imágenes candidatas. Los resultados se presentan en formato VOTable, que incluye URLs para acceder a las imágenes. La capacidad de manejar diferentes dimensiones y tipos de datos hace que SIAP sea fundamental para trabajar con grandes volúmenes de imágenes astronómicas.
- **SCS**(“Simple Cone Search”): El protocolo SCS permite recuperar registros de catálogos astronómicos a través de una consulta cónica simple. Esta consulta especifica una posición en el cielo y una distancia angular para definir un cono en el cielo. El protocolo devuelve una lista de fuentes astronómicas que se encuentran dentro de ese cono, con los resultados en formato VOTable. SCS es útil para identificar objetos astronómicos en una región específica del cielo.

- **TAP** (“Table Access Protocol”): TAP proporciona una interfaz para acceder a colecciones de datos tabulares, como catálogos astronómicos y bases de datos generales. Permite realizar consultas sobre datos expresados en forma de tablas, facilitando la búsqueda y recuperación de información estructurada. TAP es útil para trabajar con datos que se organizan en tablas y para integrar diferentes conjuntos de datos en una sola consulta, proporcionando una forma eficiente de manejar información tabular en el contexto astronómico.
- **SSA** (“Service Spectral Protocol”): Este protocolo define una interfaz uniforme para descubrir y acceder a espectros dimensionales desde colecciones de datos y metadatos. Incluye datos espectrofotométricos, series temporales, y distribuciones de energía espectral (SED), así como espectros en una dimensión. SSA permite a los usuarios recuperar información sobre la distribución de energía en función de la longitud de onda, lo que es esencial para el análisis espectral en astronomía.

Las distintas maneras de consultar información en un VO están diseñadas para ofrecer apoyo y flexibilidad a los usuarios al buscar datos. Estas metodologías de búsqueda están fundamentadas en la forma en que se organiza y se accede a la información, y se diferencian de los motores de búsqueda más complejos. Estos protocolos están diseñados para manejar datos astronómicos de una manera estructurada y especializada, adaptándose a los requisitos específicos de cada tipo de consulta.

3.2.3. PROTOCOLO SIA Y FORMATO DE IMAGENES

El estándar SIA versión 2.0, publicado oficialmente como recomendación de la IVOA en 2015 ², es una especificación diseñada para facilitar el descubrimiento, descripción y acceso a datos astronómicos en forma de imágenes y cubos multidimensionales. Este protocolo pertenece al conjunto de estándares desarrollados por la IVOA, promover la interoperabilidad de los datos astronómicos a través de plataformas virtuales y servicios web.

SIA 2.0 surge como una evolución significativa del protocolo SIA 1.0 ³, el cual se encontraba limitado al acceso a imágenes bidimensionales. Con el crecimiento del volumen, complejidad y diversidad de los datos en astronomía especialmente por el aumento de observaciones multiespectrales y multidimensionales fue necesaria una ampliación de las capacidades del protocolo para adaptarse a estos nuevos requerimientos.

SIA 2.0 está alineado con el **modelo de datos ObsCore** (Observation Core Data Model), también definido por la IVOA ⁴, que estandariza los metadatos esenciales que describen conjuntos de datos observacionales. Esta alineación garantiza una interoperabilidad robusta entre

² IVOA Documents - Simple Image Access Version 2.0

³ IVOA Documents - Simple Image Access Version 1.0

⁴ IVOA Documents - ObsCore Version 1.1

los servicios de descubrimiento y acceso, y permite a los usuarios interactuar de forma homogénea con distintos repositorios de datos astronómicos.

SIA 2.0 permite a los usuarios realizar búsquedas de imágenes o cubos de datos utilizando parámetros físicos y astronómicos estandarizados, entre los que destacan:

- **POS (posición):** define una región del cielo en coordenadas astronómicas.
- **BAND (banda):** restringe la búsqueda a un rango espectral específico (por ejemplo, radio, infrarrojo, óptico, etc.).
- **TIME (tiempo):** permite filtrar observaciones por fecha u hora de adquisición.
- **POL (polarización):** facilita el descubrimiento de datos con información sobre estados de polarización.

Estas consultas se realizan sobre **interfaces RESTful**, utilizando parámetros en las URLs. El resultado es una tabla en formato VOTable (XML estructurado para datos astronómicos), compatible con herramientas del ecosistema VO.

Gracias al uso del modelo ObsCore, los servicios que implementan SIA 2.0 pueden describir los conjuntos de datos de mejor forma. Entre los metadatos incluidos se encuentran: instrumento y telescopio utilizados, resolución espacial, espectral y temporal, tamaño angular y cobertura espectral, tipo de producto (imagen, cubo, etc.) y también DOI u otro identificador persistente. Esto permite a los astrónomos evaluar rápidamente la utilidad y relevancia científica de los datos antes de proceder a su descarga.

SIA 2.0 no solo facilita el descubrimiento de recursos, sino que también permite **obtener los archivos asociados** directamente o a través de otros servicios. Los enlaces de descarga están incluidos en los resultados de búsqueda, y pueden apuntar a archivos completos, subconjuntos de cubos de datos (por ejemplo, cortes espectrales o espaciales) e interfaces de procesamiento adicionales, como recortes o reproyecciones. Esto permite optimizar el uso de ancho de banda y almacenamiento local, ya que los usuarios pueden descargar únicamente las porciones de datos que necesitan.

El protocolo retorna principalmente archivos que contienen datos astronómicos en forma de **imágenes o cubos multidimensionales**. Los tipos de archivos más comunes que puede retornar incluyen:

- **FITS (Flexible Image Transport System):** Es el formato estándar en astronomía para el almacenamiento de imágenes, cubos de datos, espectros y tablas. Permite incluir metadatos en sus cabeceras, lo que lo hace ideal para observaciones científicas.
- **JPEG / PNG (opcional):** Algunos servicios pueden ofrecer vistas rápidas (thumbnails o versiones ligeras) en estos formatos gráficos estándar para visualización preliminar, pero no se usan para análisis científico.

- **VOTable (XML):** Aunque no es una imagen, SIAP retorna las respuestas a las consultas en formato VOTable, que contiene metadatos estructurados sobre los archivos disponibles y enlaces a los archivos descargables (normalmente FITS).

3.2.4. PYVO Y PETICIONES SIAP

PyVO es una biblioteca de Python que proporciona una interfaz para interactuar con los servicios VO, permitiendo la consulta y manipulación de datos astronómicos distribuidos. La biblioteca está diseñada para trabajar con varios protocolos estándar utilizados en el contexto de los VO, tales como VOTable, SIA, SSA, y SCS. PyVO facilita la integración con estos servicios de una manera sencilla y eficiente, proporcionando a los usuarios herramientas para obtener, procesar y analizar grandes volúmenes de datos astronómicos distribuidos a través de diferentes repositorios y bases de datos [Graham *et al.*, 2014].

Algunas de las funcionalidades clave de PyVO son:

- **Interoperabilidad:** Facilita la integración con servicios de OV que siguen los estándares IVOA.
- **Búsqueda de Datos:** Permite realizar búsquedas de datos astronómicos mediante protocolos como SCS, SIA y SSA, entre otros.
- **Manipulación de Resultados:** Ofrece soporte para el manejo de resultados en formatos como VOTables, que se utilizan para representar los resultados de las consultas de búsqueda.
- **Acceso a Imágenes y Espectros:** A través de servicios SIAP y SSA, permite la consulta y descarga de imágenes y espectros astronómicos.

Para realizar una petición SIAP utilizando PyVO, se necesita una conexión al servicio SIAP del observatorio de interés. La biblioteca pyvo permite realizar consultas a estos servicios para obtener imágenes astronómicas en formato FITS. A continuación, se explica el proceso básico para hacer una petición y convertir la imagen obtenida en un archivo PNG, como lo hicimos en el ejemplo previo.

```
1 import pyvo
2 import requests
3 from astropy.io import fits
4 import matplotlib.pyplot as plt
5
6 siap_url = "https://datalab.noirlab.edu/sia/calibrated_all"
7 service = pyvo.dal.SIAService(siap_url)
8
9 ra = 10.684
10 dec = 41.269
```

```

11 size = 0.5
12
13 results = service.search(pos=(ra, dec), size=size)
14
15 first_image_url = results[0].getdataurl()
16 print(f"URL: {first_image_url}")
17
18 response = requests.get(first_image_url)
19 fits_filename = "imagen_noirlab.fits"
20 with open(fits_filename, "wb") as f:
21     f.write(response.content)
22
23 hdu_list = fits.open(fits_filename)
24 image_data = hdu_list[0].data
25
26 plt.imshow(image_data, cmap='gray', origin='lower')
27 plt.colorbar()
28
29 png_filename = "imagen_noirlab.png"
30 plt.savefig(png_filename, format='png')

```

Código 3.1: Petición SIAP en PyVO a Registro específico.

El código [3.1](#), en python realiza una consulta a un servicio SIAP para obtener imágenes astronómicas a partir de **coordenadas celestes específicas**. En primer lugar, se define la URL del servicio SIAP, que corresponde al archivo de datos astronómicos del observatorio NOIRLab. A continuación, se configura la búsqueda de imágenes utilizando las coordenadas de ascensión recta (RA) y declinación (Dec), además de especificar el tamaño del campo de visión deseado. Luego, el código realiza la consulta al servicio y obtiene la URL de la imagen FITS más cercana a las coordenadas solicitadas. Esta imagen se descarga a través de una solicitud HTTP usando el módulo requests. Finalmente, el archivo FITS descargado se lee utilizando la librería Astropy, que permite procesar los datos astronómicos en este formato, y luego se convierte en una imagen visualizable en formato PNG mediante el uso de matplotlib. Esta conversión permite visualizar la imagen astronómica en un formato más accesible y ampliamente utilizado en plataformas web y de análisis.

Además de realizar consultas directas a servicios específicos mediante protocolos como SIAP, los Observatorios Virtuales permiten descubrir recursos astronómicos a través de **registros (Registry)**, los cuales actúan como catálogos públicos que describen los servicios disponibles, sus tipos de datos, cobertura espacial, y protocolos soportados. Esta funcionalidad es especialmente útil cuando no se conoce de antemano la URL de un servicio o se desea explorar qué recursos existen para un tipo específico de consulta. Utilizando el módulo pyvo.registry, es posible buscar servicios según criterios como su tipo (por ejemplo, imágenes, espectros o catálogos), su nombre, o el estándar que implementan, como SIAP, SSAP o TAP. Esta búsqueda basada en metadatos permite identificar servicios compatibles con las necesidades del usuario y acceder a ellos posteriormente para realizar consultas más específicas. De esta forma, el uso del registro se convierte en una herramienta fundamental para descubrir y acceder a datos astronómicos de manera flexible y estandarizada.

```
1 import pyvo
2
3 servicios_siap = pyvo.registry.search(servicetype='sia')
4 for servicio in servicios_siap[]:
5     print("Título:", servicio.title)
6     print("URL de acceso:", servicio.access_url)
7     print("Descripción:", servicio.short_description)
```

Código 3.2: Búsqueda en el registro de los servicios que implementan SIAP

El código 3.2, también en python, permite buscar servicios astronómicos que ofrecen imágenes mediante el protocolo SIAP, sin necesidad de conocer sus URLs previamente. A partir de los resultados, se puede elegir un servicio y utilizarlo para realizar consultas con coordenadas.

3.3 DESDE IA A PROCESAMIENTO DE LENGUAJE NATURAL

3.3.1. INTRODUCCIÓN A LA INTELIGENCIA ARTIFICIAL (IA)

La IA es una rama de la informática que busca desarrollar sistemas capaces de realizar tareas que, tradicionalmente, requieren de la inteligencia humana, tales como el reconocimiento de voz, la toma de decisiones y la comprensión del lenguaje natural.

Un hito fundamental en la conceptualización de la IA fue el trabajo de **Alan M. Turing**. En 1950, Turing publicó el artículo *Computing Machinery and Intelligence* [Turing, 1950]. En este trabajo, planteó la pregunta: *¿Pueden las máquinas pensar?*, y propuso reemplazar esta cuestión por una más concreta y verificable mediante un experimento que denominó el **Juego de Imitación**, conocido como el **Test de Turing**.

El Test de Turing consiste en una conversación a ciegas entre un evaluador humano y dos participantes: una máquina y un ser humano. Si el evaluador no puede distinguir cuál de los dos es la máquina basándose únicamente en las respuestas proporcionadas, se considera que la máquina ha demostrado una forma de inteligencia comparable a la humana.

Turing argumentó que, en lugar de debatir sobre la definición de *pensar* aplicada a las máquinas, era más práctico evaluar si una máquina podía imitar con éxito el comportamiento humano en una conversación. Este enfoque pragmático sentó las bases para futuras investigaciones en IA, enfocándose en la capacidad de las máquinas para exhibir comportamientos inteligentes observables. Además, en su artículo, abordó diversas objeciones a la posibilidad de máquinas pensantes y discutió la viabilidad de crear máquinas que aprendan y se auto-mejoren, anticipando el aprendizaje automático moderno. La visión de Turing ha tenido una influencia perdurable en el campo de la IA, estableciendo un marco para evaluar el progreso en la creación de sistemas capaces de emular aspectos de la cognición humana.

3.3.2. APRENDIZAJE AUTOMÁTICO

El aprendizaje automático (Machine Learning) es una subárea de la IA que permite a los sistemas aprender patrones a partir de datos sin estar explícitamente programados. A diferencia de los enfoques tradicionales, en los que los algoritmos siguen instrucciones predefinidas, el aprendizaje automático capacita a las máquinas para identificar y generalizar patrones en los datos, lo que les permite tomar decisiones y realizar predicciones sobre nuevos datos basándose en experiencias pasadas.

De manera más formal, [Mitchell, 1997] define el Aprendizaje Automático de la siguiente manera:

“Se dice que un programa aprende de una experiencia E con respecto a una tarea T y medida de desempeño P si el desempeño en la tarea T medida por P mejora con la experiencia E ”.

El aprendizaje automático se clasifica generalmente en tres tipos principales:

- **Aprendizaje supervisado:** En este enfoque, los algoritmos aprenden a partir de un conjunto de datos etiquetado, donde cada entrada tiene una salida asociada (o etiqueta). El modelo busca la relación entre las entradas y salidas para poder predecir la salida de nuevas entradas no vistas. Ejemplos típicos de tareas supervisadas incluyen la clasificación y la regresión.
- **Aprendizaje no supervisado:** En el aprendizaje no supervisado, los algoritmos trabajan con datos no etiquetados y buscan patrones o estructuras ocultas en los datos. Este tipo de aprendizaje es útil cuando no se dispone de etiquetas o se desea descubrir grupos o relaciones dentro de los datos. Un ejemplo común es el análisis de clusters, donde los datos se agrupan según sus características similares.
- **Aprendizaje por refuerzo:** En este enfoque, los algoritmos aprenden a través de la interacción con un entorno, recibiendo recompensas o penalizaciones por sus acciones. El objetivo es aprender una política que maximice la recompensa acumulada a lo largo del tiempo. Este tipo de aprendizaje es común en la robótica y en aplicaciones como los videojuegos y los vehículos autónomos.

En el ámbito del aprendizaje automático, los modelos son la clave. Estos modelos son entrenados utilizando datos de entrada para realizar tareas específicas, como clasificación, regresión o agrupación. Un modelo bien entrenado puede realizar tareas complejas.

3.3.3. REDES NEURONALES ARTIFICIALES

Las **Redes Neuronales Artificiales (ANN)** son modelos computacionales inspirados en el funcionamiento del cerebro humano, específicamente en la forma en que las neuronas biológicas se conectan y procesan información. Este enfoque marca un punto de inflexión dentro del aprendizaje automático, al permitir modelar relaciones no lineales complejas entre variables de entrada y salida.

El origen teórico de las redes neuronales se remonta al trabajo de Warren McCulloch y Walter Pitts [McCulloch y Pitts, 1943], que propusieron el primer modelo de una **neurona artificial**. En este trabajo, describieron una neurona como una unidad binaria capaz de recibir múltiples entradas, aplicar una función de activación lógica, y producir una única salida. Este modelo fue un intento pionero por formalizar computacionalmente cómo el cerebro podría realizar cálculos lógicos a través de redes de neuronas interconectadas. Aunque simple, sentó las bases de lo que más adelante sería conocido como neuronas artificiales.

Una red neuronal artificial consiste en un conjunto de nodos (neuronas artificiales) como se pueden ver en la imagen 3.3.

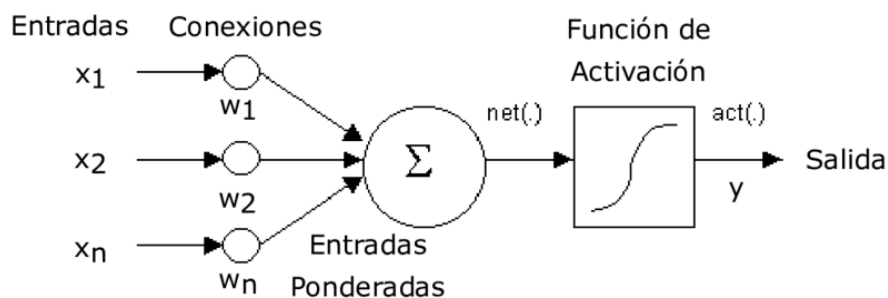


Figura 3.3: Estructura de una neurona artificial

Fuente: [ResearchGate](#).

Estos nodos están organizados en capas, dada conexión entre neuronas tiene un peso que se ajusta durante el proceso de entrenamiento para minimizar el error del modelo. Las funciones de activación (como ReLU, sigmoide o tanh) introducen no linealidades, permitiendo a la red aprender relaciones complejas.

Aunque los primeros modelos neuronales eran prometedores, tenían limitaciones importantes, como la incapacidad para resolver problemas no lineales (demostrada por Minsky y Papert [Minsky y Papert, 1969] con el perceptrón).

Décadas después, un avance clave fue introducido por Geoffrey Hinton [Rumelhart et al., 1986]. Este paper fue fundamental porque introdujo el algoritmo de **backpropagation** como un método eficiente para entrenar redes neuronales multicapa. Además, Hinton propuso el concepto de **representación distribuida**, donde el conocimiento

no está localizado en una sola neurona, sino distribuido a través de muchas, lo que permite a la red generalizar y aprender conceptos más abstractos. Esto abrió la puerta a modelos más profundos y poderosos.

3.3.4. PROCESAMIENTO DE LENGUAJE NATURAL

El **Procesamiento de Lenguaje Natural (NLP)** es una disciplina dentro de la inteligencia artificial que se centra en la interacción entre las computadoras y el lenguaje humano. Su objetivo es permitir que las máquinas comprendan, interpreten y generen texto de manera que resulte significativa y útil. A lo largo de las décadas, el NLP ha experimentado una evolución significativa, marcada por diversos enfoques y avances tecnológicos.

En la década de los 90s los modelos de lenguaje buscaban predecir la **probabilidad** de una secuencia de palabras en un idioma. Se popularizaron los modelos de n-gramas, que estimaban la probabilidad de una palabra basada en las $n-1$ palabras anteriores, como [Brown *et al.*, 1992]. Aunque efectivos en ciertos contextos, los modelos de n-gramas tenían limitaciones, especialmente en la captura de dependencias a largo plazo en el texto. A principios de la década de 2000, se introdujeron modelos neuronales para el **procesamiento del lenguaje**. Yoshua Bengio hizo un Modelo de Lenguaje Neuronal Probabilístico [Bengio *et al.*, 2003], que utilizaba una red neuronal para aprender representaciones distribuidas de palabras y capturar dependencias de largo alcance en el texto. Este enfoque superó a los modelos tradicionales de n-gramas y sentó las bases para futuros desarrollos en NLP.

En 2017, el artículo de Vaswani, introdujo el modelo **Transformer** [Vaswani *et al.*, 2017], que hizo un gran avance el NLP al permitir el procesamiento paralelo de secuencias y capturar dependencias a largo plazo de manera más eficiente que las **redes recurrentes (RNN)**. Este modelo se convirtió en la base para desarrollos posteriores en modelos de lenguaje a gran escala. El Transformer es una arquitectura que reemplaza las redes neuronales recurrentes y convolucionales (CNN) utilizando únicamente mecanismos de atención. En este modelo, el **encoder** procesa la secuencia de entrada y la transforma en representaciones continuas, mientras que el **decoder** genera la secuencia de salida basándose en estas representaciones, utilizando la atención para alinear y enfocar la información relevante. Este enfoque mejora la paralelización, reduce significativamente el tiempo de entrenamiento y, a la vez, incrementa la calidad en tareas de transducción secuencial como la traducción automática.

La figura 3.4 ilustra la arquitectura general del Transformer, dividiéndola en dos partes principales: el encoder y el decoder. En la parte izquierda, el encoder toma una secuencia de **entrada** y, mediante capas apiladas de auto-atención y capas totalmente conectadas aplicadas de manera puntual, la transforma en una secuencia de representaciones continuas. En la parte derecha, el decoder genera la secuencia de **salida** de forma autoregresiva, lo que significa que, en cada paso, utiliza no solo las representaciones de la secuencia de entrada producidas por el encoder, sino también los símbolos ya generados previamente para producir el siguiente símbolo. Esta estructura modular y apilada permite procesar la información

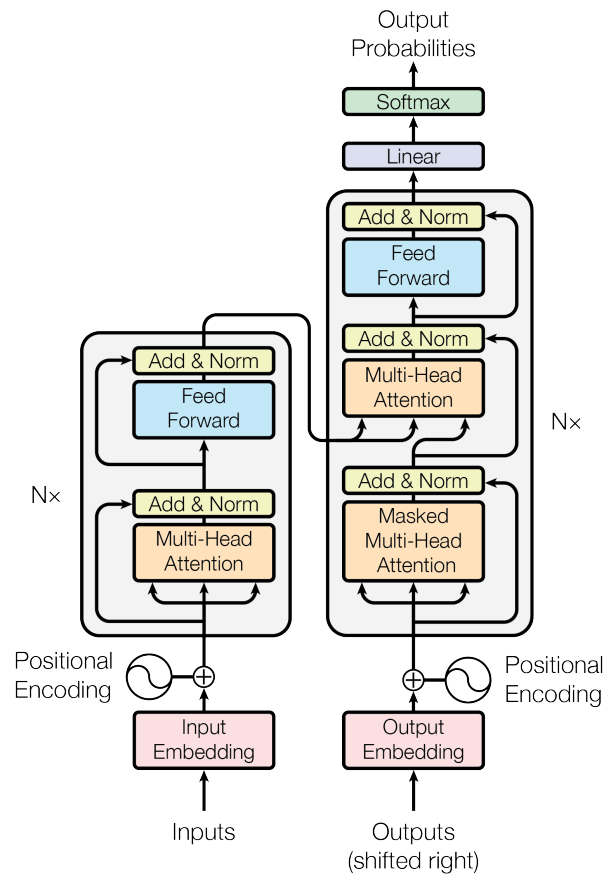


Figura 3.4: Arquitectura del Transformer.

Fuente: [Vaswani et al., 2017]

de forma paralela y eficiente, optimizando el rendimiento.

3.4 CHATGPT y DEEPSEEK

3.4.1. OPENAI: GPT Y O1

El desarrollo de modelos de lenguaje ha experimentado una evolución acelerada en la última década, y entre estos, ChatGPT de **OpenAI** se destaca por su capacidad para mantener conversaciones coherentes y contextuales. ChatGPT es el producto culminante de una serie de avances en modelos preentrenados de lenguaje, basados en la arquitectura Transformer, que han transformado el campo del Procesamiento de Lenguaje Natural.

La historia de ChatGPT se remonta a la introducción de la serie **Generative Pre-trained Transformer (GPT)**. El primer modelo, GPT-1, presentó la idea de preentrenar un modelo en grandes corpus de texto para luego ajustarlo a tareas específicas [Radford et al., 2018]. Los auto-

res introdujeron un enfoque innovador para el procesamiento del lenguaje natural basado en dos etapas: primero, un **preentrenamiento generativo** de un modelo de lenguaje sobre grandes corpus no etiquetados, utilizando una arquitectura Transformer unidireccional para aprender representaciones generales del idioma mediante la predicción de la siguiente palabra; y segundo, un **ajuste fino discriminativo**, donde el mismo modelo es adaptado a tareas específicas (como clasificación o respuesta a preguntas) con una pequeña cantidad de datos etiquetados y mínimas modificaciones en la arquitectura. Esta estrategia demostró superar significativamente a modelos entrenados exclusivamente para cada tarea, estableciendo un nuevo paradigma en NLP al evidenciar la eficacia del aprendizaje auto-supervisado y la transferencia de conocimiento entre tareas.

Esta idea fue ampliada significativamente en GPT-2, que, con 1.500 millones de parámetros, demostró que el aumento de escala en los modelos generativos conduce a una mejora sustancial en la coherencia y diversidad del texto generado [Radford et al., 2019]. El salto evolutivo se hizo aún más notable con GPT-3, que, con 175.000 millones de parámetros, consolidó la capacidad de los modelos para realizar tareas de lenguaje con muy poca o ninguna adaptación específica [Brown et al., 2020].

Con el éxito de GPT-3, OpenAI lanzó ChatGPT en 2022, ajustándolo específicamente para el diálogo interactivo. Esta versión se entrenó para mantener contextos conversacionales extendidos, permitiendo respuestas más naturales y coherentes en comparación con versiones previas. Además, la evolución continua ha llevado a la aparición de GPT-4 y variantes como GPT-4o, que incorporan capacidades multimodales al procesar tanto texto como imágenes, ampliando el espectro de aplicaciones y la precisión en la generación de contenido [Achiam et al., 2023].

OpenAI ha extendido su impacto al ofrecer una **API** robusta que permite a desarrolladores integrar capacidades avanzadas de procesamiento del lenguaje natural en sus aplicaciones. La API de OpenAI proporciona acceso a los modelos de la serie GPT, incluyendo las versiones optimizadas para tareas conversacionales, como ChatGPT, permitiendo que los desarrolladores realicen desde simples consultas hasta complejas tareas de generación de contenido, análisis de texto y traducción.

OpenAI ofrece una variedad de modelos a través de su API ⁵, cada uno diseñado para satisfacer diferentes necesidades en tareas de procesamiento de lenguaje natural y generación de contenido. A continuación, se presenta una lista de los principales modelos disponibles:

- **GPT-4.5 Preview:** El modelo más grande y capaz de OpenAI hasta la fecha, diseñado para tareas complejas que requieren un razonamiento avanzado y generación de texto de alta calidad.
- **o3-mini:** Un modelo rápido y flexible, optimizado para razonamiento inteligente en áreas como programación, matemáticas y ciencias. Es una opción rentable para apli-

⁵OpenAI Platform

caciones que requieren eficiencia y precisión.

- **GPT-4o**: Un modelo multimodal capaz de analizar y generar texto, imágenes y sonido. Ofrece un rendimiento rápido e inteligente, adecuado para aplicaciones que requieren procesamiento de múltiples tipos de datos.
- **OpenAI o1**: Un modelo de razonamiento de vanguardia que admite herramientas, salidas estructuradas y visión. Diseñado para resolver problemas complejos y de múltiples pasos.
- **OpenAI o3-mini**: Un modelo de razonamiento pequeño y rentable, optimizado para programación, matemáticas y ciencias. Admite herramientas y salidas estructuradas.

Los modelos de la serie **o** se destacan por su capacidad avanzada de razonamiento y la generación de salidas estructuradas, diferenciándose de los modelos GPT tradicionales que se enfocan principalmente en la generación de texto fluido. Estos modelos están diseñados para abordar tareas complejas que requieren múltiples pasos de inferencia, integrando herramientas y utilizando longitudes de contexto extendidas para procesar grandes volúmenes de información. En esencia, los modelos **o** no solo generan texto, sino que también aplican un razonamiento lógico más profundo, lo que los hace especialmente adecuados para aplicaciones en las que se requiere una solución detallada y coherente a problemas complejos. El documento de lanzamiento de los modelos OpenAI o1 y o1-mini [Jaech et al., 2024], explica que estos modelos se entrenan mediante aprendizaje por refuerzo a gran escala para incorporar una cadena de pensamiento, lo que mejora sus capacidades de razonamiento. Esta metodología permite que los modelos razonen sobre las políticas de seguridad en contexto, lo que se traduce en un rendimiento de vanguardia orientados a mitigar riesgos como la generación de consejos ilícitos, caer en respuestas estereotipadas y vulnerabilidades.

3.4.2. DEEPSEEK-V2 Y DEEPSEEK-R1

DeepSeek nació en 2023 como una spin-off de un exitoso fondo de cobertura chino fundado por Liang Wenfeng en 2015. En 2021, antes de que EE.UU. impusiera restricciones a la exportación de chips avanzados a China, Liang acumuló 10,000 GPUs de **Nvidia** (modelos para entrenar modelos de IA ⁶). Este movimiento estratégico permitió a DeepSeek desarrollar tecnología puntera a pesar de las limitaciones geopolíticas.

En enero de 2024, se publicó [Bi et al., 2024] este artículo introduce el primer modelo de la compañía, **DeepSeek LLM**, con 67 mil millones de parámetros. El estudio se centra en las leyes de escalamiento y presenta hallazgos que facilitan la ampliación de modelos de lenguaje de código abierto. Además, mediante técnicas de **ajuste fino supervisado (SFT)** y **optimización directa de preferencias (DPO)**, se han derivado los modelos DeepSeek Chat,

⁶WIRED: Qué hay detrás de DeepSeek

que demuestran un rendimiento sobresaliente. Evaluaciones abiertas revelan que DeepSeek LLM 67B Chat exhibe un rendimiento superior en comparación con GPT-3.5, especialmente en tareas de código, matemáticas y razonamiento, ver figura 3.5.

Model	STEM	Humanities	Reasoning	Coding	Math	Extraction	Roleplay	Writing	Average
GPT-4-1106-preview*	9.90	9.95	8.10	9.05	7.95	9.90	9.50	9.70	9.26
GPT-3.5-turbo-0613*	9.55	9.95	6.20	7.05	7.05	9.00	8.65	9.65	8.39
LLAMA-2-Chat 7B*	8.65	8.75	4.25	3.00	2.40	6.50	7.70	8.90	6.27
LLAMA-2-Chat 13B*	8.63	9.75	5.10	3.00	3.45	6.93	7.50	8.85	6.65
LLAMA-2-Chat 70B*	8.93	9.63	5.80	3.15	3.30	7.25	7.50	9.30	6.86
Zephyr-Beta 7B*	9.03	9.63	5.60	5.10	4.45	7.45	8.20	9.35	7.35
Xwin 70b v0.1*	9.68	9.95	6.55	4.25	3.30	8.75	8.25	9.55	7.53
Xwin 13b v0.2*	9.55	9.88	5.20	3.60	2.85	7.70	8.60	8.68	7.01
TÜLU 2+DPO 70B*	9.00	9.90	7.00	4.70	4.65	9.35	9.25	9.25	7.89
DeepSeek LLM 67B Chat	9.60	9.70	8.00	7.35	6.25	8.40	8.20	9.30	8.35
DeepSeek LLM 67B Chat DPO	9.70	9.80	9.05	6.75	6.65	9.30	9.10	9.75	8.76

Figura 3.5: Evaluación MT-Bench DeepSeek LLM.

Fuente: [Bi et al., 2024]

En mayo de 2024, se publica el artículo [Liu et al., 2024a] donde se presenta **DeepSeek-V2**, un modelo **Mixture-of-Experts (MoE)** con 236 mil millones de parámetros totales. En el artículo se presenta un modelo de lenguaje MoE que combina entrenamiento económico e inferencia eficiente. El modelo cuenta con 236 mil millones de parámetros en total. DeepSeek-V2 incorpora innovaciones arquitectónicas como **Multi-head Latent Attention (MLA)**, que comprime significativamente la caché Key-Value, para entrenar modelos robustos a un costo reducido. Comparado con DeepSeek 67B, DeepSeek-V2 demuestra un rendimiento significativamente superior, ahorrando un 42,5 % en costos de entrenamiento, reduciendo la caché KV en un 93,3 %. En resumen, con esto se optimiza memoria, uso de GPU y energía.

En diciembre de 2024, se presenta el artículo de **DeepSeek-V3** [Liu et al., 2024b], un modelo MoE con 671 mil millones de parámetros totales. El modelo introduce estrategias innovadoras para el **balanceo de carga** y establece un objetivo de entrenamiento de predicción de múltiples tokens para mejorar el rendimiento. DeepSeek-V3 ha demostrado un rendimiento comparable al de modelos líderes en la industria, manteniendo una eficiencia notable en términos de recursos computacionales.

En enero de 2025, se publica **DeepSeek-R1** [Guo et al., 2025]. En este trabajo se presentan los modelos de razonamiento de primera generación, DeepSeek-R1-Zero y DeepSeek-R1. Inicialmente, se entrenó DeepSeek-R1-Zero mediante **aprendizaje por refuerzo** a gran escala sin utilizar ajuste fino supervisado previo, lo que permitió que el modelo desarrollara, de forma natural, comportamientos de razonamiento potentes e interesantes. Sin embargo, se identificaron problemas en su legibilidad y en la mezcla de idiomas. Para abordar estas limitaciones y potenciar aún más las capacidades de razonamiento, se introdujo DeepSeek-R1, que incorpora un entrenamiento en múltiples etapas y el uso de datos de arranque en frío antes de aplicar el aprendizaje por refuerzo. Como resultado, DeepSeek-R1 logró un rendimiento comparable al de OpenAI-o1 en tareas de razonamiento. Además, para apoyar a la comunidad investigadora, se han puesto a disposición los modelos DeepSeek-R1-Zero, DeepSeek-R1,

y seis modelos densos destilados de DeepSeek-R1.

3.5 CHATBOTS CON LLM

El desarrollo de **chatbots** y **agentes** con inteligencia artificial ha experimentado un gran crecimiento estos últimos años, impulsado por los avances en modelos de lenguaje a gran escala y técnicas de aprendizaje profundo. Según [Arsenijević y Jović, 2019], estos agentes con IA se caracterizan por su capacidad para interactuar con los usuarios de manera natural, tomar decisiones y ejecutar tareas de forma autónoma.

3.5.1. PROMPTS

Un elemento clave en el diseño y desempeño de estos agentes es el **prompt engineering**, la ciencia de diseñar las entradas o instrucciones (prompts) que se le entregan a los modelos de lenguaje. Un prompt bien formulado puede guiar al modelo para generar respuestas más precisas, coherentes y relevantes, lo que resulta esencial en contextos donde la interacción debe ser fluida y contextualizada. Esta técnica ha ganado relevancia a medida que se descubrió que la calidad del output de un modelo puede depender críticamente de la forma en que se le plantea la pregunta o el problema. En este sentido, un estudio de Ye, respalda esta idea al mostrar que diseñar prompts efectivos es una tarea desafiante que requiere pensar cuidadosamente sobre los errores del modelo y mejorar la forma en que se le pide la información [Ye et al., 2023]. Los autores proponen un método llamado PE2, que añade tres elementos clave a la hora de crear el prompt:

- Descripciones detalladas.
- Una buena especificación del contexto.
- Una plantilla que guíe el razonamiento paso a paso.

Con estos ajustes, el **método PE2** logra mejorar los resultados en diversas tareas (por ejemplo, superando enfoques previos en ciertos problemas en un 6,3 % y 3,1 %, y en otros hasta un 6,9 %). Esto demuestra que, al mejorar cómo se estructura el prompt, se puede lograr que el modelo responda de manera más acertada y adaptada a tareas complejas, lo que es esencial para el éxito de chatbots y otros agentes de inteligencia artificial.

Otro artículo de Giray [Giray, 2023], explica que un prompt se compone de varios elementos clave:

- Primero, está la instrucción, que es la tarea específica o indicación que guía el comportamiento del modelo para lograr el resultado deseado.

- Luego, se encuentra el contexto, que añade información externa o de fondo que ayuda al modelo a entender mejor la situación y generar respuestas más precisas.
- El tercer elemento es la entrada de datos, que es la pregunta o información central que se quiere que el modelo procese.
- Finalmente, el indicador de salida define el tipo o formato en que se quiere que se presente la respuesta, ya sea un resumen, un ensayo, una lista, etc.

En conjunto, estos componentes permiten comunicar de manera efectiva nuestras intenciones al modelo.

3.5.2. EVALUACIONES

En relación con la evaluación de la inteligencia de estos modelos, investigaciones recientes han demostrado que los Large Language Models (LLM), en muchos casos, pueden alcanzar niveles de desempeño que les permiten superar el **Test de Turing** [Jones y Bergen, 2025]. Esto significa que, en determinadas condiciones y contextos, estos modelos son capaces de mantener conversaciones tan naturales y coherentes que un evaluador humano no puede distinguir si la respuesta proviene de una máquina o de un ser humano. Dicho logro subraya el potencial de los LLM en tareas de generación de lenguaje, pero también plantea desafíos éticos y de seguridad en su implementación.

Ahora si el objetivo es evaluar el desempeño de un chatbot o agente, existen múltiples estrategias, y la elección del enfoque depende del objetivo específico de la aplicación. Algunas **métricas** se enfocan en la calidad lingüística de la respuesta (fluidez, coherencia, relevancia), mientras que otras están orientadas a tareas más estructuradas como recuperación de información, clasificación o razonamiento.

ROUGE (Recall-Oriented Understudy for Gisting Evaluation), es una métrica ampliamente utilizada en tareas de resumen automático y generación de texto [Lin, 2004]. Compara el texto generado por el modelo con uno o más textos de referencia, analizando coincidencias en n-gramas, secuencias de palabras y estructuras. Cuanto mayor sea la coincidencia, mejor será considerada la salida del modelo. Su uso es particularmente útil cuando se busca evaluar qué tan bien un chatbot o modelo de lenguaje puede generar respuestas informativas, coherentes y similares a las que daría un experto humano. En el caso de sistemas conversacionales, puede servir para validar si las respuestas entregadas son coherentes con un conjunto de respuestas esperadas.

Por otro lado existe **Recall@K** (ver fórmula 3.1), esta métrica es común en sistemas de recomendación y recuperación de información, y se aplica en escenarios donde el modelo entrega un conjunto de K posibles resultados. En resumen, mide si el resultado correcto se encuentra dentro de esos K resultados. Un valor alto de Recall@K indica que el sistema

es eficaz recuperando respuestas útiles dentro de sus primeras opciones. En agentes inteligentes que consultan bases de datos o servicios externos, permite evaluar la capacidad del sistema para entregar resultados relevantes entre sus primeras respuestas, lo cual es crucial para la eficiencia de uso.

$$\text{Recall@K} = \frac{\text{Número de elementos relevantes en el top } K}{\text{Número total de elementos relevantes disponibles}} \quad (3.1)$$

MRR (Mean Reciprocal Rank) que es otra métrica relacionada a la recuperación de información, que evalúa la posición del primer resultado relevante entre los entregados por el modelo. Se calcula como el recíproco del ranking del primer resultado correcto y se promedia entre muchas pruebas (ver fórmula 3.2). Esta métrica premia los sistemas que logran ubicar respuestas útiles en las primeras posiciones, lo que es especialmente importante en sistemas donde el orden de las respuestas influye directamente en la experiencia del usuario.

$$\text{MRR} = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\text{rank}_i} \quad (3.2)$$

Donde, $|Q|$ es el Número total de consultas) y rank_i es la Posición del primer resultado relevante para la i -ésima consulta.

3.5.3. IMPACTO SOCIAL Y AMBIENTAL

En un contexto de creciente preocupación por el impacto social y ético de estas tecnologías, trabajos recientes destacan la necesidad de evaluar a ChatGPT y otros modelos bajo el enfoque **SPADE (Sustainability, Privacy, Digital divide, and Ethics)**. Este enfoque propone examinar no solo el rendimiento técnico de los modelos, sino también su sostenibilidad ambiental, la protección de la privacidad de los usuarios, la reducción de la brecha digital y la consideración de implicaciones éticas en su despliegue.

A medida que estos modelos de IA se integran en aplicaciones críticas, es fundamental implementar evaluaciones rigurosas que aborden estos aspectos para asegurar una adopción responsable y equitativa de la tecnología [Khowaja *et al.*, 2024].

Se hace mucho incapié en el lado **ambiental**, se menciona que desde la popularización de ChatGPT en noviembre de 2022, el tema de su sostenibilidad ambiental ha ganado relevancia, aunque no ha sido ampliamente discutido en comparación con otros aspectos de su desarrollo. Los modelos de lenguaje grandes, como ChatGPT, tienen un alto impacto ambiental debido al considerable consumo de energía durante el proceso de entrenamiento. Este proceso requiere el uso de grandes volúmenes de datos y el ajuste de billones de parámetros, lo que conlleva un alto costo en términos de **huella de carbono**.

La huella de carbono de estos modelos está directamente relacionada con las fuentes de energía utilizadas para alimentar los centros de datos. Las fuentes renovables, como la energía solar o hidroeléctrica, tienen una huella baja, mientras que las fuentes como el carbón y el gas natural resultan en emisiones más altas. La **sostenibilidad** en el contexto de los LLMs debe abordar tanto el uso eficiente de la energía como el uso de fuentes renovables. Optimizar el hardware y los algoritmos de entrenamiento puede ayudar a reducir el impacto ambiental. A medida que la tecnología de IA avanza, es crucial que los desarrolladores implementen estrategias que minimicen el impacto ecológico y promuevan un equilibrio entre el progreso tecnológico y la responsabilidad ambiental.

3.6 HERRAMIENTAS DE DESARROLLO WEB

3.6.1. STACK PARA DESARROLLO DE PLATAFORMAS

El desarrollo de una **plataforma web** para implementar chatbots y agentes conversacionales requiere una base tecnológica sólida y coherente. Elegir un stack tecnológico definido no solo garantiza la integración fluida de todas las herramientas y servicios necesarios, sino que también acelera el desarrollo, facilita el mantenimiento y mejora la escalabilidad de la aplicación. **MERN Stack** es una excelente elección para el desarrollo de plataformas debido a su capacidad para ofrecer un rendimiento óptimo y escalabilidad, sus componentes son:

- **MongoDB:** Una base de datos NoSQL orientada a documentos que permite almacenar datos en formato flexible (similar a JSON). Su escalabilidad y facilidad para gestionar grandes volúmenes de datos la hacen ideal para aplicaciones que manejan información dinámica y sin un esquema fijo.
- **Express.js:** Un framework minimalista para Node.js que simplifica la creación de aplicaciones web y APIs, ofreciendo una estructura ágil y escalable para el desarrollo del back-end.
- **React.js:** Una biblioteca para construir interfaces de usuario interactivas y dinámicas. Su enfoque basado en componentes facilita la reutilización de código y permite el desarrollo de aplicaciones web que responden a las necesidades del usuario de forma eficiente.
- **Node.js:** Un entorno de ejecución para JavaScript que permite ejecutar código del lado del servidor. Gracias a su modelo de I/O asíncrono y no bloqueante, Node.js es ideal para aplicaciones en tiempo real y de alta escalabilidad.

El artículo [Lakshmi *et al.*, 2023] destaca la importancia de mantener una **latencia** baja, ya que asegura una respuesta rápida y una experiencia fluida para el usuario. Al trabajar con tecnologías basadas en JavaScript tanto en el front-end (React) como en el back-end (Node.js

y Express) y almacenar datos de manera flexible con MongoDB, se logra una integración homogénea y eficiente que facilita la comunicación entre componentes. Esto no solo permite implementar aplicaciones complejas de forma más ágil, sino que también ayuda a mantener una velocidad de respuesta alta, lo cual es crucial para entornos donde cada milisegundo cuenta. Además, la popularidad y el amplio soporte de la comunidad del MERN Stack facilitan la resolución de problemas y la incorporación de mejoras, asegurando que la plataforma se mantenga competitiva a lo largo del tiempo.

3.7 AGENTE ASTRONÓMICO

En el marco de un proyecto de memoria de título, se desarrolló un asistente virtual de tipo chatbot destinado al Observatorio Virtual [Álvarez, 2025]. Este asistente se presenta como un agente inteligente muy completo, basado en un Modelo de Lenguaje Grande (LLM), que cuenta con acceso a herramientas capaces de recuperar, transformar y proporcionar información relevante al usuario. El sistema fue diseñado con una arquitectura que incluye tanto un grafo de procesamiento de datos como diversas herramientas especializadas que facilitan la interacción del modelo con los usuarios.

El agente fue construido utilizando **LangChain**, una librería que permite crear aplicaciones de lenguaje natural mediante la integración de Modelos de Lenguaje Grande con fuentes de datos externas y herramientas de procesamiento, facilitando la creación de flujos de trabajo complejos. LangChain facilita que el agente pueda interactuar con diversas bases de datos y servicios de manera fluida. Para la obtención de datos astronómicos, se utilizó PyVO para interactuar con los observatorios virtuales y realizar consultas a través de protocolos estándar como TAP (Table Access Protocol), SIAP (Simple Image Access Protocol), SLAP (Simple Line Access Protocol), SCS (Simple Cone Search) y SSA (Simple Spectral Access). Estos protocolos permiten acceder a vastos conjuntos de datos astronómicos, desde imágenes hasta datos espectrales.

La interfaz gráfica fue diseñada para que los usuarios pudieran interactuar de manera sencilla con el sistema, realizando consultas sobre datos astronómicos, las cuales eran procesadas y respondidas por el agente. Se utilizó la API de **GPT-4o-mini**, una versión optimizada del modelo GPT-4, para generar las respuestas del agente de manera más rápida.

Posteriormente, se realizaron pruebas con usuarios de diferentes niveles de conocimiento astronómico para evaluar el rendimiento del agente. Estas pruebas permitieron obtener datos sobre el funcionamiento del sistema, el cumplimiento de los objetivos establecidos y las sugerencias para su mejora. El análisis de los resultados indicó que el asistente fue útil y logró despertar el interés de los usuarios, quienes reconocieron su potencial. Sin embargo, se concluyó que el agente necesita más iteraciones y mejoras para cumplir completamente con su objetivo de acercar la astronomía a la población general y facilitar el acceso a la información para los profesionales del área.

Capítulo 4

PROPUESTA DE SOLUCIÓN

Para comparar el desempeño de los modelos de inteligencia artificial generativa *GPT-4o* y *DeepSeek-R1*, se diseñará e implementará una interfaz de prueba que integre ambos modelos, permitiendo realizar consultas en tiempo real y comparar los resultados obtenidos. A partir de esta plataforma, se evaluará su eficacia en términos de rapidez y precisión en el contexto de búsqueda científica, especialmente en la recuperación de imágenes astronómicas.

En este contexto, se desarrollará **VOAI-Search**, una interfaz web que funcionará como entorno de evaluación para los modelos mencionados. Esta plataforma operará como un chatbot accesible desde el navegador, donde el usuario podrá interactuar con *GPT-4o* y *DeepSeek-R1* de forma independiente, enviando consultas en lenguaje natural relacionadas con la búsqueda de imágenes astronómicas. La interfaz permitirá seleccionar el modelo con el cual se desea trabajar, ejecutar consultas sobre servicios SIAP y visualizar los resultados devueltos por dichos servicios.

Además de su propósito experimental, VOA-Search tendrá el potencial de evolucionar hacia un sistema más robusto, como un agente inteligente o asistente conversacional especializado en astronomía, capaz de facilitar el acceso a datos científicos mediante lenguaje natural.

4.1 ARQUITECTURA GENERAL

La arquitectura general del sistema, representada en la Figura 4.1, se compone de seis elementos principales interconectados, cada uno cumpliendo un rol específico dentro de la plataforma *VOAI-Search*:

- **Frontend:** corresponde a la interfaz gráfica accesible desde el navegador por parte del usuario. Desde aquí se redactan y envían las consultas en lenguaje natural, y se visualizan los resultados devueltos por el sistema.

- **Backend:** actúa como intermediario entre el frontend y los demás módulos. Su función es coordinar el flujo de información, reenviar las consultas a los modelos de lenguaje y al motor de búsqueda, y administrar la conexión con la base de datos.
- **Base de datos:** almacena información relevante del sistema como logs, historial de interacciones, consultas realizadas y resultados obtenidos, permitiendo tanto trazabilidad como evaluación posterior.
- **Motor de búsqueda astronómica:** es un componente especializado que interpreta las instrucciones generadas por los modelos de lenguaje natural y ejecuta consultas en servicios astronómicos utilizando protocolos estándar, como SIAP.
- **Modelos de lenguaje natural:** están representados por las APIs de *GPT* y *DeepSeek*, a las cuales el sistema envía las consultas del usuario. Cada modelo responde con una interpretación de la consulta en formato estructurado, generalmente una query compatible con servicios VO.
- **Servicio de Observatorios Virtuales (VO):** es el destino final de las consultas astronómicas. Aquí se accede a colecciones de datos científicos, principalmente imágenes astronómicas, las cuales son recuperadas por el motor de búsqueda para su visualización.

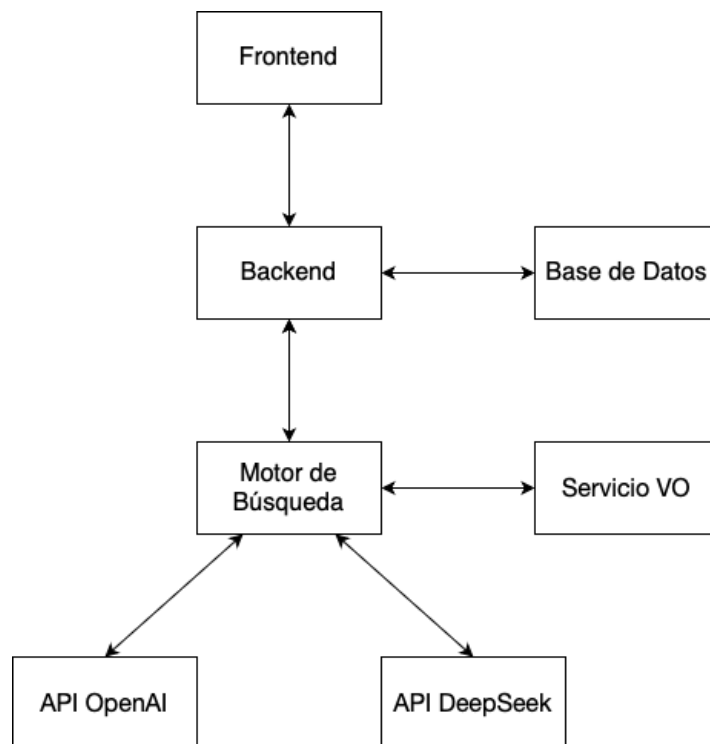


Figura 4.1: Diagrama de la Arquitectura de VOAI-Search.
Fuente: Elaboración propia.

4.1.1. DESCRIPCIÓN DEL STACK TECNOLÓGICO

Para el desarrollo de la plataforma VOAI-Search se optará por el stack MERN (MongoDB, Express, React, Node.js), cuya combinación de tecnologías basadas en JavaScript garantiza un ciclo de desarrollo ágil, un alto grado de modularidad y una integración nativa con APIs RESTful. Este enfoque facilita la creación de una interfaz de prueba rápida que permita comparar los dos modelos de lenguaje (*GPT-4o* y *DeepSeek-R1*) mediante sus respectivas APIs.

En primer lugar, el **frontend** se implementará con React.js y Material UI (MUI). React permite construir componentes reutilizables y gestionar el estado de la aplicación de manera declarativa, mientras que MUI aporta un conjunto de componentes visuales estilizados que aceleran la maquetación y aseguran una experiencia de usuario consistente y responsiva. Esta combinación resulta ideal para prototipar rápidamente un chat de evaluación, donde cada mensaje, botón o vista de resultados se construye como un componente independiente y testeable.

En el **backend**, Node.js y Express proporcionarán un servidor ligero y altamente concurrente. Node.js, por su naturaleza orientada a eventos, gestiona de forma eficiente las múltiples peticiones simultáneas que generan las consultas a los modelos y al motor de búsqueda astronómica. Express, por su parte, ofrece un enrutamiento sencillo y middleware configurable, lo que permitirá implementar rápidamente las rutas para:

- Recepción y validación de consultas desde el frontend.
- Reenvío de prompts a las APIs de *GPT-4o* y *DeepSeek-R1*.
- Orquestación del flujo hacia el microservicio de consultas VO.
- Persistencia y recuperación de logs, historiales y resultados en la base de datos.

El **motor de consultas VO** se desarrollará como un microservicio independiente en Python, basándose en PyVO y FastAPI. PyVO simplifica la ejecución de protocolos estándar como SIAP, mientras que FastAPI ofrece un rendimiento elevado y validación automática de esquemas JSON. De este modo, el motor recibirá las instrucciones estructuradas por los modelos de lenguaje, ejecutará las consultas astronómicas y devolverá los enlaces o metadatos resultantes de manera uniforme al backend.

La **base de datos** será MongoDB, una solución NoSQL que almacena documentos JSON de forma nativa. Esto encaja perfectamente con las estructuras de datos que intercambian el backend, los modelos de lenguaje y el motor VO, evitando transformaciones innecesarias y permitiendo escalabilidad horizontal. Además, MongoDB facilita la creación de índices sobre campos de consulta (por ejemplo, coordenadas o identificadores de experimento), mejorando el rendimiento en el análisis posterior de métricas.

En conjunto, el stack MERN permite desarrollar y desplegar VOAI-Search con rapidez y flexibilidad, aprovechando la homogeneidad de JavaScript en cliente y servidor, la robustez de las bibliotecas para interfaces reactivas y la capacidad de integrar sin fricción microservicios especializados. Este enfoque resulta especialmente adecuado para experimentar con distintos modelos de lenguaje a través de APIs, ya que cada componente puede evolucionar independientemente y escalar según la carga de trabajo generada durante las pruebas comparativas.

4.2 DISEÑO Y DESARROLLO DE FRONTEND

Antes de comenzar con el desarrollo del frontend, se definió una propuesta de diseño centrada en los principios de usabilidad, claridad visual y accesibilidad, con el objetivo de crear una interfaz para que los usuarios pudieran interactuar de forma sencilla con los modelos de lenguaje natural evaluados en este estudio. VOAI-Search fue concebido como una plataforma web de tipo chatbot, enfocada en reducir la fricción en el proceso de formular consultas complejas a observatorios virtuales mediante lenguaje natural.

4.2.1. IDENTIDAD VISUAL

El logo del sistema VOAI-Search fue diseñado con una intención clara de representar la unión entre el Observatorio Virtual y la Inteligencia Artificial, conceptos que dan origen al acrónimo VOAI. Este diseño se presenta con el texto *VOAI Research*, destacando el enfoque investigativo del sistema y su vínculo con el análisis de datos astronómicos mediante técnicas de IA.

Como podemos ver en la figura 4.2, el elemento más distintivo del logo se encuentra en la letra O, que fue reemplazada por una lupa que contiene en su interior una luna y estrellas. Este símbolo refuerza visualmente la noción de exploración, búsqueda e investigación astronómica. Además, la lupa como ícono se asocia naturalmente con interfaces de búsqueda o investigación, lo que potencia la claridad del propósito del sistema para usuarios nuevos.

El estilo del logo sigue una estética casi minimalista, con formas limpias y fácilmente reconocibles. Se trabajaron distintas combinaciones de color para adaptarse a diversos contextos visuales, manteniendo siempre una buena legibilidad y coherencia con la paleta general de la aplicación. La tipografía utilizada también fue seleccionada para transmitir modernidad y tecnología sin perder legibilidad.

Este logo no solo cumple una función visual decorativa, sino que también refuerza la identidad del sistema y su propósito: facilitar el acceso a datos astronómicos complejos mediante herramientas conversacionales basadas en inteligencia artificial.

La paleta de colores utilizada cumple un rol central en este diseño, no solo en términos estéticos, sino como una herramienta funcional que mejora la experiencia de uso, especialmente



Figura 4.2: Logo VOA AI Search.

Fuente: Elaboración propia.

en un entorno orientado a la interacción conversacional y al análisis visual de resultados.

La paleta base está compuesta principalmente por tres colores: un azul oscuro (#122736), un blanco puro (#FFFFFF) y un naranja vibrante (#f58220). El azul oscuro fue elegido como color de fondo para la aplicación y la interfaz del chatbot, ya que reduce la fatiga visual y favorece el enfoque prolongado, lo cual es relevante al tratarse de una plataforma que puede ser utilizada durante sesiones extensas de consulta y exploración. El blanco se emplea para los textos sobre este fondo oscuro, asegurando un alto nivel de contraste y máxima legibilidad, incluso en condiciones de baja luminosidad. Por su parte, el color naranja actúa como color de acento o acción, aplicándose exclusivamente a elementos clave como botones, enlaces y respuestas destacadas, garantizando una rápida identificación de acciones posibles por parte del usuario.

Desde el punto de vista de accesibilidad, esta combinación cumple con los estándares de contraste establecidos por las pautas WCAG 2.1 (Web Content Accessibility Guidelines) ¹, asegurando que los elementos sean fácilmente distinguibles por personas con discapacidades visuales leves o en pantallas con bajo brillo. De esta forma, la plataforma mantiene una

¹<https://www.w3.org/WAI/standards-guidelines/wcag/glance/es>

accesibilidad visual adecuada sin comprometer la estética del diseño. Finalmente, la elección de un azul oscuro como color base permite una compatibilidad natural con el modo oscuro, cada vez más común en aplicaciones modernas. A diferencia del negro puro, el azul oscuro proporciona una experiencia visual más suave, evitando contrastes agresivos y facilitando una lectura prolongada sin fatiga ocular.

4.2.2. INTERFAZ

En cuanto a la interfaz de usuario desarrollada para VOAI-Search, esta se compone principalmente de componentes React organizados de forma modular para facilitar su mantenimiento y escalabilidad. Dentro del directorio `/components`, se incluyen piezas clave como `Chat.jsx`, `Message.jsx`, `InputBox.jsx`, `Sidebar.jsx` y `Header.jsx`. Estas piezas se ensamblan en las páginas principales contenidas en el directorio `/pages`, particularmente `ChatPage.jsx`, encargada de orquestar toda la lógica visual y de interacción del sistema.

La interfaz presentada en las figuras [4.3](#) y [4.4](#) corresponde a dos versiones del sistema: una vista en formato escritorio y otra en formato móvil. En ambas se puede observar la visualización de una imagen astronómica en formato FITS —en este caso, una imagen de la estrella Sirius—, la cual es retornada como resultado de una consulta al chatbot. En la parte inferior de la pantalla, se mantiene visible la caja de entrada de texto, acompañada de un botón de envío para las preguntas astronómicas. El diseño es completamente responsive, adaptándose a diferentes tamaños de pantalla mediante el uso de flexbox y media queries implementadas con la librería Material UI (MUI).

En términos funcionales, se ha implementado un sistema de scroll automático para mantener la vista centrada en los últimos mensajes, lo cual mejora la usabilidad en sesiones largas de conversación. Además, los mensajes se distinguen visualmente según su origen: los mensajes del usuario se alinean a la derecha con un color de fondo personalizado, mientras que los mensajes del chatbot se muestran alineados a la izquierda, acompañados del ícono del modelo utilizado.

Un elemento clave de la interfaz es el botón que permite alternar entre los dos modelos de lenguaje natural que serán evaluados en esta investigación: GPT-4o y DeepSeek R1. Este botón permite realizar comparaciones controladas al mantener el mismo prompt y observar cómo varían las respuestas según el modelo seleccionado. Esta funcionalidad es fundamental para los experimentos que buscan comparar precisión, tiempo de respuesta, y otros indicadores de desempeño.

La estructura del frontend se organiza principalmente en el directorio `/frontend`, que contiene carpetas y archivos clave para el correcto funcionamiento de la aplicación web. Entre ellos, destacan la carpeta `src`, donde reside el código fuente principal, y la carpeta `public`, que almacena recursos estáticos como imágenes de iconos y logos.

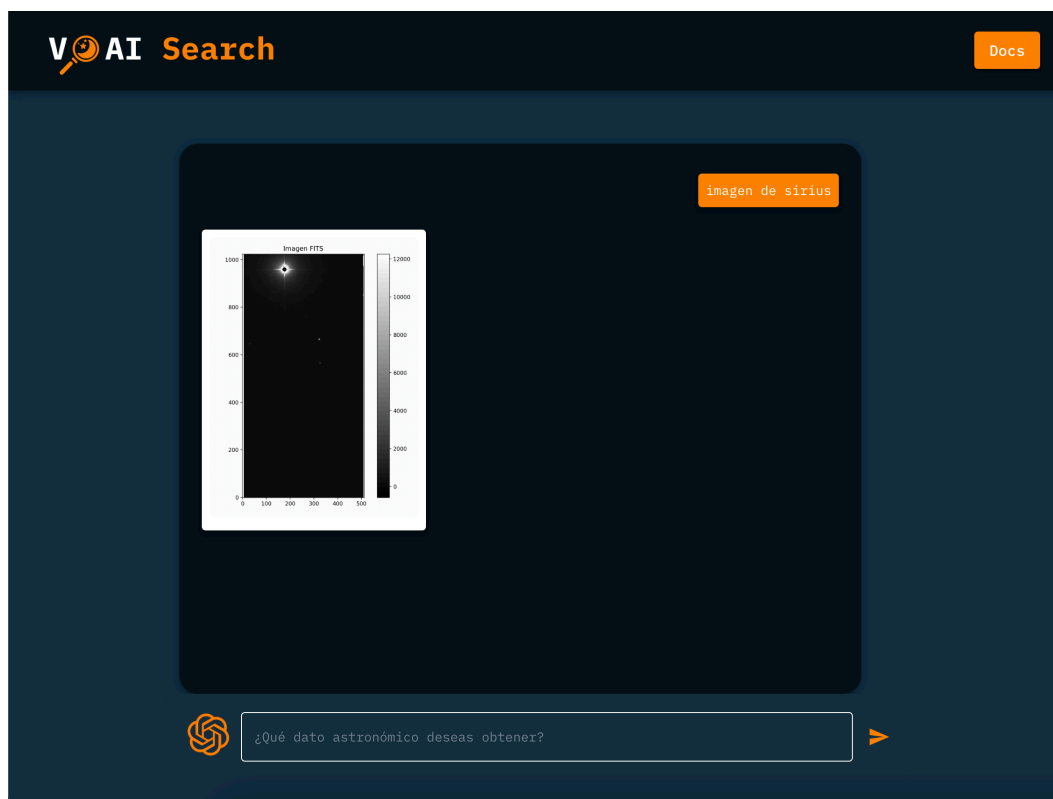


Figura 4.3: Interfaz de escritorio.
Fuente: Elaboración propia.

El código fuente está dividido en componentes que cumplen funciones específicas para construir la interfaz y manejar la interacción con el usuario. Entre los principales componentes se encuentran:

- **parentComponent:** Este es el componente contenedor principal que gestiona el estado global de la aplicación, incluyendo el historial de mensajes, la interacción con el backend y la coordinación entre los diferentes elementos de la interfaz. Actúa como el núcleo de la aplicación React, orquestando la comunicación y actualización de datos entre los demás componentes.
- **chat:** Encargado de mostrar la conversación entre el usuario y lo que retornan las APIs de IA. Este componente renderiza una lista de mensajes utilizando múltiples instancias del componente message.
- **message:** Componente responsable de representar un mensaje individual dentro del chat, sea este un mensaje enviado por el usuario o una respuesta generada por el backend. Permite mostrar el contenido de manera clara y diferenciada.
- **inputBox:** Proporciona el campo de entrada donde el usuario puede escribir sus consultas o mensajes. Se encarga de capturar el texto ingresado y enviarlo al componente

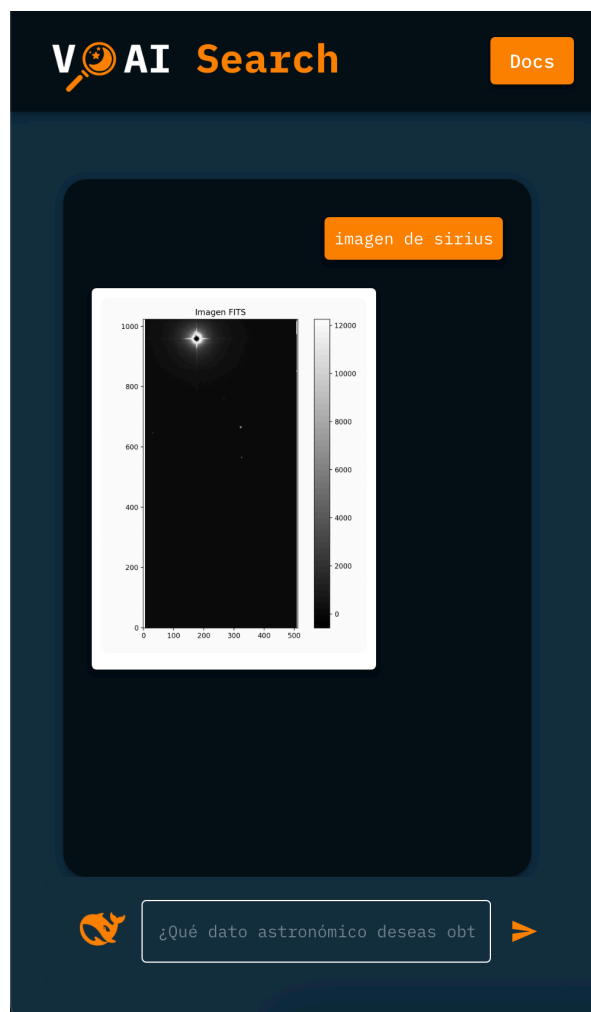


Figura 4.4: Interfaz movil.
Fuente: Elaboración propia.

contenedor para su procesamiento.

- **header:** Representa la barra superior de la aplicación, donde se muestra información como el título, logo y elementos de navegación (Documentación).
- **ailcon:** Es un componente visual que funciona como un ícono o botón relacionado con las APIs de IA, con este botón podemos intercambiar entre una consulta a la API de OpenAI o DeepSeek.

La comunicación entre estos componentes está centralizada en el parentComponent, que mantiene el estado de la conversación y maneja las llamadas a la API del backend para obtener las respuestas a las consultas del usuario. Esta arquitectura basada en componentes favorece la modularidad, facilitando la mantenibilidad y escalabilidad del frontend.

En conjunto, la combinación de React y Vite, junto con una estructura clara y modular de componentes, permite ofrecer una experiencia de usuario fluida y responsiva en la aplicación VOAI-Search, facilitando la interacción con la inteligencia artificial mediante una interfaz intuitiva y eficiente.

4.3 DISEÑO Y DESARROLLO DE BACKEND

El backend de la plataforma VOAI-Search fue desarrollado utilizando `Node.js` junto con el framework `Express.js`, permitiendo una arquitectura modular, escalable y de rápido desarrollo. La lógica principal se centró en diseñar una API RESTful capaz de interactuar con modelos de lenguaje natural, registrar mensajes, y almacenar las interacciones generadas por los usuarios dentro del sistema, manteniendo siempre una estructura clara, desacoplada y fácilmente mantenible. A continuación, se describe el diseño y funcionamiento de sus componentes principales.

El punto de entrada del servidor es el archivo `server.js`, en el cual se configura el entorno de ejecución, la conexión a la base de datos, las rutas del sistema y el middleware necesario para procesar las solicitudes. Se utiliza el paquete `dotenv` para gestionar variables de entorno como la URI de conexión a MongoDB y la configuración de los puertos del servidor, lo que mejora la portabilidad del sistema entre entornos de desarrollo, prueba y producción, a la vez que asegura la separación entre código y datos sensibles.

4.3.1. BASE Y MODELO DE DATOS

Como sistema de almacenamiento se empleó `MongoDB Atlas`, una solución de base de datos NoSQL basada en documentos que ofrece alta flexibilidad para representar estructuras de datos no relacionales como los mensajes de chat. Esta elección responde a la necesidad de almacenar conversaciones de manera eficiente, sin depender de esquemas rígidos, permitiendo así una evolución progresiva del modelo de datos. La conexión a la base se realiza mediante la librería `mongoose`, que actúa como una capa de modelado de objetos (ODM), facilitando la definición de esquemas y la validación de datos.

La URI de conexión es almacenada de forma segura en el archivo `.env`, lo que permite que el sistema acceda a la base de datos sin exponer credenciales directamente en el código fuente. Este enfoque mejora la seguridad y facilita el despliegue del sistema en distintos entornos, como servidores locales o servicios en la nube.

Para el almacenamiento de las conversaciones entre los usuarios y los modelos de lenguaje, se definió un único modelo de datos denominado `Chat`. Este modelo permite representar cada mensaje como un documento independiente dentro de la colección correspondiente en MongoDB, registrando atributos claves para reconstruir y analizar la interacción completa

del usuario con el sistema.

El esquema Chat fue diseñado utilizando `mongoose.Schema` y contempla los siguientes campos:

- `text`: campo de tipo `String` que almacena el contenido textual del mensaje.
- `sender`: campo de tipo `String` que identifica si el mensaje fue enviado por el user o por el backend.
- `aiIcon`: campo que especifica el modelo al cual se le hizo la consulta (`openai` o `deepseek`).
- `timestamp`: campo de tipo `Date` que registra el momento exacto en el que el mensaje fue creado, permitiendo mantener el orden temporal de las interacciones.

Este diseño permite almacenar tanto las peticiones del usuario como las respuestas generadas por los modelos LLM, posibilitando la reconstrucción detallada de cada conversación. Además, al incluir metadatos como el identificador del modelo, se puede realizar un análisis posterior de desempeño, permitiendo comparar la efectividad de distintos modelos de lenguaje natural en tareas de recuperación de información astronómica.

4.3.2. RUTAS Y LÓGICA DE API

La API expone endpoints específicos para manejar las operaciones relacionadas con los chats. En particular, la ruta `/api/chats` permite realizar operaciones `GET` para obtener todos los mensajes almacenados y `POST` para registrar un nuevo mensaje en la base de datos. Estas operaciones están definidas en un archivo de rutas separado (`chatRoutes.js`), lo que promueve la separación de responsabilidades y facilita la extensión del sistema en el futuro.

Además, se implementó un middleware personalizado para el manejo de errores y la validación de entradas, garantizando que los datos almacenados mantengan una estructura coherente y que cualquier fallo durante el procesamiento de las solicitudes sea manejado adecuadamente. La respuesta de cada endpoint se retorna en formato `JSON`, lo que facilita su integración con aplicaciones frontend como la interfaz de chat implementada en `React`.

Las rutas fueron divididas en archivos separados ubicados en la carpeta `routes/`. Esto permite mantener un diseño modular, facilitando su mantenimiento y escalabilidad futura. Para esta instancia solo se creó la ruta de chats:

- `chatRoutes`: Maneja el registro y obtención de mensajes del chat. Se conecta al modelo Chat.

Cada ruta expone métodos GET y POST, lo cual permite al frontend tanto recuperar información como registrar nuevas interacciones.

Desde su concepción, el backend fue diseñado pensando en su extensión a futuro. La separación de modelos, rutas y lógica central permite integrar fácilmente nuevas funcionalidades, como autenticación de usuarios, historiales de búsqueda personalizados o soporte para nuevos modelos de lenguaje. Además, el uso de `Express.js` permite integrar middlewares para validaciones, control de errores, y autenticación sin afectar el núcleo del sistema.

En conjunto, este backend provee una base sólida y extensible para el desarrollo de herramientas interactivas de consulta en astronomía basadas en lenguaje natural, manteniendo una arquitectura moderna, desacoplada y fácil de escalar.

4.4 INTEGRACIÓN CON MODELOS LLM

La integración con modelos LLM constituye el núcleo de la plataforma VOAI-Search, ya que a través de ellos se traduce la expresión en lenguaje natural de un usuario en consultas astronómicas estructuradas, diseñadas para ser ejecutadas directamente contra servicios de observatorios virtuales. Gracias a esta capa, todo el conocimiento técnico sobre protocolos IVOA y formatos de archivo queda oculto tras un simple chat, facilitando la interacción a usuarios sin experiencia previa en astronomía.

El propósito fundamental de conectar la plataforma con modelos LLM fue demostrar que éstos pueden asistir de manera fiable en la construcción de consultas científicas complejas. En lugar de requerir que el usuario formule manualmente parámetros de posición, banda espectral o formato de salida, basta con plantear la pregunta en términos cotidianos. El LLM se encarga de identificar el tipo de recurso deseado (imágenes, espectros, listado de objetos, líneas espectrales o descubrimiento de servicios) y de generar un objeto de datos intermedio que contiene todos los campos necesarios para realizar la petición al servicio IVOA adecuado. Este objeto intermedio se representa en formato JSON y se compone de claves como:

- `category`: el protocolo IVOA elegido (en este trabajo solo se probará 'SIA').
- `object`: identificador o nombre del objeto astronómico.
- `posAR` y `posDEC`: coordenadas en grados de ascensión recta y declinación.
- `size`: tamaño del campo de búsqueda en grados.
- `bandpass`: banda espectral deseada.
- `instrument` y `server`: metadatos adicionales para afinar la búsqueda.

Para cada interacción el backend realiza dos llamadas diferenciadas a la API de LLM:

1. **Clasificación de la consulta:** Se envía el mensaje original del usuario junto a un prompt de sistema que solicita únicamente un JSON con la categoría IVOA y los metadatos necesarios. Esta llamada determina si se utilizará SIA, SSA, SCS, SLA o REGISTRY, y extrae nombre(s) de objeto e información preliminar.
2. **Optimización de parámetros SIA:** Una vez clasificada la consulta como SIA, se realiza una segunda llamada donde se reenvían las coordenadas y parámetros básicos obtenidos, junto con un prompt que pide optimizar el tamaño del campo, la banda y priorizar servicios. La respuesta es otro JSON que contiene valores refinados y una lista de URL de servicios recomendados.

4.4.1. LÓGICA DEL MOTOR DE BÚSQUEDA

El motor de búsqueda, implementado íntegramente en Python, actúa como el corazón de la plataforma, orquestando desde la clasificación inicial hasta la entrega de la imagen final. Al recibir una petición HTTP en el único endpoint expuesto, el sistema carga las variables de entorno y los módulos especializados (PyVO, Astroquery, Astropy, PIL y Matplotlib), y arranca el flujo de trabajo en dos fases claramente diferenciadas.

En la primera fase se invoca una función de clasificación que envía al modelo de lenguaje que el usuario seleccionó desde el frontend, un mensaje de sistema diseñado para extraer únicamente un objeto JSON con los parámetros fundamentales de la consulta. El prompt de sistema tiene la siguiente forma:

```
'You are an expert assistant in Astronomy and Virtual  
Observatories. Analyze the user's query and respond with a JSON  
object that includes the resource category (category, which can  
be SIA for images, SSA for spectra, SCS for object search, SLA for  
spectral lines, or REGISTRY for service discovery) and a list  
of names or identifiers for the astronomical object (object,  
including SIMBAD designations such as Messier numbers, NGC/IC  
numbers, common names, or other catalog identifiers). If no  
object is mentioned, return an empty list; if the query is not  
related to a specific object, return a random object from SIMBAD.  
Additionally, include title (dataset or image title), format (file  
format), instrument (instrument used), posAR (Right Ascension  
coordinates), posDEC (Declination coordinates), bandpass (band  
used in the observation), and server (specific IVOA server if  
required). If no applicable information is available, return  
'none'. Respond only with the JSON object, without explanations.'
```

A continuación, esta llamada retorna un bloque JSON como el siguiente ejemplo:

```
{ 'category': 'SIA',  
  'object': ['M51'],  
  'posAR': 202.484,  
  'posDEC': 47.2306,  
  'size': 0.01,  
  'bandpass': 'nir',  
  'instrument': '2MASS',  
  'server': 'https://irsa.ipac.caltech.edu' }
```

Este JSON inicial determina que se utilizará el protocolo SIAP y proporciona los valores básicos para posición, banda y servidor. Si alguna coordenada aparece como 'none', el motor resuelve automáticamente el valor consultando el servicio Vizier y transformando los resultados con Astropy a un formato compatible con IVOA.

En la segunda fase, y solo si la categoría es SIA, se realiza una nueva llamada al modelo para *optimizar* los parámetros de búsqueda. El prompt para optimización de SIA es:

```
'You are an expert in astronomy data retrieval. Given the  
following parameters for a SIA query (IVOA), optimize them to  
increase the chance of finding an image. Respond with a JSON  
containing 'size' (recommended search size in degrees), 'bandpass'  
(suggested alternative if none is found), and 'priority_services'  
(a list of recommended SIA service URLs). Search for images in the  
specified format, bandpass, and instrument. Respond just with the  
JSON object, without explanations.'
```

La respuesta de optimización devuelve por ejemplo:

```
{ 'size': 0.05,  
  'bandpass': 'red',  
  'priority_services': ['https://jvo.nao.ac.jp'] }
```

Este bloque JSON se valida sintáctica y semánticamente, comprobando que los campos existen y que sus tipos son correctos, antes de pasar a la ejecución.

Finalmente, para construir la consulta definitiva el motor utiliza la librería PyVO para descubrir todos los servicios SIA disponibles, ordenándolos según la lista de `priority_services` sugerida por el modelo y barajando el resto con `random.shuffle()`. Al localizar una URL o imagen FITS válida, el sistema descarga el fichero, lo abre con Astropy, extrae la primera extensión de datos, genera un gráfico en escala de grises con Matplotlib y guarda la imagen resultante en formato PNG. Luego se retorna la ruta o la URL de la imagen al cliente,

completando así un flujo modular y extensible que separa claramente la responsabilidad de generación de parámetros del LLM de la ejecución de búsquedas astronómicas.

La abstracción en dos llamadas a la API de LLM y la separación del motor de búsqueda en un módulo independiente permiten añadir nuevos modelos, protocolos o servicios sin afectar la lógica central. Además, facilita el enrutamiento dinámico según criterios de coste, latencia o historial de rendimiento.

4.5 FLUJO DE DATOS

Resumiendo el flujo de procesos detallando cómo una consulta de usuario viaja desde la interfaz hasta la obtención de los datos astronómicos y la respuesta final, haciendo énfasis en las dos llamadas a los modelos LLM y en la interacción con el servicio SIAP.

1. **Envío desde el Frontend.** El usuario ingresa un mensaje en la interfaz de chat implementada en React, que encapsula el texto en un objeto JSON y hace un POST a `/api/chats` para guardar el mensaje y a la ruta principal de ejecución (`/execute`) para procesarlo.
2. **Registro inicial en el Backend.** Al recibir la petición, el servidor Express almacena el mensaje de usuario en MongoDB mediante el modelo `Chat`, registrando campos como `text`, `sender='user'`, `aiIcon` y `timestamp`.
3. **Desencadenamiento del Motor de Búsqueda.** A continuación, el backend envía el mismo texto al servicio Python a través de un POST al endpoint `/execute`, iniciando el motor de búsqueda astronómica.
4. **Primera llamada a LLM (Clasificación).** El motor Python empaca un prompt de clasificación. Este prompt y el mensaje de usuario se envían al modelo seleccionado (OpenAI o DeepSeek).
5. **Resolución de coordenadas (si es necesario).** Si el JSON inicial contiene `'posAR': 'none'` o `'posDEC': 'none'`, el motor invoca el servicio Vizier mediante Astroquery para traducir el nombre del objeto en coordenadas, transformándolas con Astropy a un par (`ra`, `dec`) en grados.
6. **Segunda llamada a LLM (Optimización SIA).** Con los parámetros básicos confirmados, el motor envía un segundo prompt, donde el mensaje de usuario incluye, `RA`, `DEC`, `size`, `format`, `bandpass` y `instrument`. Este bloque se valida y se integra en la búsqueda definitiva.
7. **Ejecución de la consulta SIAP.** Utilizando PyVO, el motor recupera la lista de servicios SIA disponibles. Prioriza aquellos indicados en `priority_services` y mezcla el resto aleatoriamente con `random.shuffle()`. Para cada servicio, realiza la búsqueda con

los parámetros optimizados y selecciona la primera imagen FITS o URL de una imagen válida.

8. **Procesamiento de la imagen FITS.** Al obtener una URL FITS, el motor la descarga, la abre con Astropy, extrae la primera extensión con datos de imagen, genera un gráfico en escala de grises con Matplotlib y guarda el resultado como PNG. En el caso de que la URL sea de una imagen PNG, se guarda el link directamente.
9. **Respuesta al Backend.** El motor retorna un objeto JSON con la clave `bot_response` que contiene la URL o la ruta del PNG generado. El backend Express guarda este mensaje como un nuevo documento Chat con `sender= 'bot '` y `aiIcon` correspondiente.
10. **Entrega al Frontend.** Finalmente, el backend envía la respuesta al frontend, donde se muestra la imagen o el enlace en la interfaz de chat, completando el ciclo de interacción para el usuario.

4.6 MÉTRICAS DE EVALUACIÓN

Para comparar el desempeño de los modelos de lenguaje natural GPT-4o y DeepSeek-R1 en el contexto de consultas a observatorios virtuales, se utilizarán múltiples métricas que evalúan distintos aspectos del sistema: desde la relevancia de las respuestas generadas, hasta su eficiencia, costo computacional y huella ambiental. Estas métricas permiten una evaluación más integral del comportamiento de cada modelo en tareas científicas reales.

4.6.1. RECALL@K

La métrica Recall@K evalúa si una respuesta considerada correcta aparece entre las K primeras opciones retornadas por el sistema. En esta evaluación, sin embargo, se hace una adaptación importante: una respuesta se considera correcta si el sistema logra retornar al menos una imagen, independientemente de su calidad visual o nivel de detalle.

Esta decisión se basa en que el enfoque principal no es evaluar la imagen en sí, sino verificar si el modelo logró transformar correctamente la consulta en lenguaje natural en parámetros válidos para los servicios astronómicos (específicamente SIAP). Cuando el sistema no retorna ninguna imagen, se interpreta como un fallo del modelo para entender correctamente la intención de búsqueda y parametrizarla adecuadamente para los servicios externos.

Se emplearán valores de $K = 1$ y $K = 3$, y la métrica será registrada para cada consulta y promediada por ítem temático. La ecuación correspondiente se presenta en la Ecuación [3.1](#).

4.6.2. MRR

MRR (Mean Reciprocal Rank) mide la posición de la **primera respuesta relevante** en una lista ordenada de resultados, calculando el promedio de la inversa del ranking de esa respuesta en múltiples consultas. Esta métrica es especialmente útil cuando se espera que el sistema entregue una respuesta útil en las primeras posiciones.

En el contexto de esta evaluación, una respuesta se considera relevante si el sistema logra retornar al menos una imagen astronómica como resultado. Es decir, no se evalúa la calidad visual ni la resolución de la imagen retornada, sino si el modelo fue capaz de entregar correctamente los parámetros necesarios para que el sistema completo (incluyendo los servicios SIAP) obtenga una imagen. Si no se retorna ninguna imagen, se interpreta como un fallo en el razonamiento o comprensión semántica del modelo. La fórmula utilizada se muestra en la fórmula 3.2.

4.6.3. TIEMPO DE RESPUESTA

Mide el tiempo total (en segundos) que tarda el sistema en procesar una consulta desde que el usuario la envía hasta que recibe una respuesta completa, ya sea una imagen o un mensaje indicando que no se encontró ningún resultado.

Este tiempo incluye toda la cadena de procesamiento del sistema completo: desde el frontend donde se envía el prompt, pasando por el backend (incluyendo el script en Python que se comunica con la API del modelo de lenguaje), hasta el tiempo que tarda ese mismo backend en utilizar los parámetros generados por el modelo para realizar las consultas a los servicios SIAP y obtener una imagen astronómica. Por lo tanto, esta métrica refleja tanto la velocidad del modelo de lenguaje como la eficiencia de integración con los servicios astronómicos externos.

4.6.4. COSTO POR CONSULTA

Dado que tanto GPT-4o como DeepSeek-R1 pueden tener costos asociados por el uso de sus API (por ejemplo, costos por cada millón de tokens procesados), se registrará el costo monetario de cada consulta. Esto permitirá evaluar la viabilidad económica de cada modelo en aplicaciones a gran escala.

Para cada llamada a un modelo, se calculará el costo estimado utilizando la siguiente expresión:

$$C = \left(\frac{t_{\text{in}}}{10^6} \cdot p_{\text{in}} \right) + \left(\frac{t_{\text{out}}}{10^6} \cdot p_{\text{out}} \right) \quad (4.1)$$

donde t_{in} y t_{out} representan la cantidad de tokens de entrada y salida respectivamente, mientras que p_{in} y p_{out} son las tarifas monetarias por millón de tokens de entrada y salida (en USD).

Las tarifas en el momento que se escribe esto:

- GPT-4o (OpenAI)²:
 - $p_{\text{in}} = \$5,00$ por millón de tokens
 - $p_{\text{out}} = \$20,00$ por millón de tokens
- DeepSeek-R1 (Horario estándar UTC 00:30–16:30)³:
 - $p_{\text{in}} = \$0,27$ por millón de tokens
 - $p_{\text{out}} = \$1,10$ por millón de tokens

La fórmula 4.1 se aplicará en cada consulta para registrar el `costo_estimado` como parte de los metadatos del sistema, permitiendo comparaciones objetivas entre modelos según su eficiencia económica, además de su rendimiento técnico.

4.6.5. HUELLA DE CARBONO ESTIMADA POR CONSULTA

Cada vez que se realiza una consulta a un modelo de lenguaje, se procesan tokens que requieren energía computacional, lo que conlleva una emisión estimada de CO₂. Aunque estas emisiones pueden parecer pequeñas a nivel individual, su impacto se vuelve significativo cuando se consideran millones de consultas diarias a nivel global.

Según el análisis de *DitchCarbon*⁴, el costo ambiental de los modelos de lenguaje varía dependiendo de factores como la arquitectura, el hardware usado y el tamaño del modelo. A partir de estos datos, se proponen dos fórmulas para estimar la huella de carbono por consulta en base al número de tokens generados:

- Para modelos de OpenAI (GPT-4o, GPT-o3):

Se estima que modelos de gran escala (288B+ parámetros) procesando en GPU NVIDIA H100 en precisión FP16 generan aproximadamente 30 mg de CO₂ por token:

²<https://openai.com/api/pricing/>

³https://api-docs.deepseek.com/quick_start/pricing

⁴<https://ditchcarbon.com/blog/llm-carbon-emissions>

$$\text{CO}_2^{\text{OpenAI}}(\text{mg}) = 30 \times T \quad (4.2)$$

donde T es el número de tokens de output generados por consulta.

■ **Para modelos DeepSeek (DeepSeek-R1):**

Dado que DeepSeek-R1 utiliza una arquitectura *Mixture-of-Experts* (MoE), con eficiencia computacional mejorada, se estima una emisión de 15 mg de CO₂ por token:

$$\text{CO}_2^{\text{DeepSeek}}(\text{mg}) = 15 \times T \quad (4.3)$$

Esta estimación considera las optimizaciones de hardware y arquitectura detalladas en [Liu et al., 2024a, Guo et al., 2025], y se alinea con los valores intermedios reportados por DitchCarbon para modelos eficientes sobre A100 o aceleradores similares.

Estas fórmulas permiten cuantificar el impacto ambiental de cada consulta realizada y promueven un uso consciente de modelos de lenguaje, incorporando criterios de sostenibilidad en el análisis comparativo.

4.6.6. ROUGE

ROUGE es una familia de métricas comúnmente utilizada para evaluar tareas de generación de texto. Se emplea principalmente en sistemas que generan respuestas textuales. En este proyecto, ROUGE podría ser útil si el sistema estuviera retornando respuestas basadas en texto, como descripciones de las imágenes o resúmenes de las consultas. Sin embargo, dado que en este experimento se está trabajando con SIAP (un sistema que retorna solo imágenes o enlaces a imágenes astronómicas), **ROUGE no se utilizará**. No obstante, ROUGE sigue siendo una excelente opción para evaluar un agente o chatbot que interactúe con respuestas textuales, ya que permite medir la calidad y similitud del texto generado.

4.7 PLANIFICACIÓN DE PRUEBAS Y PROCESAMIENTO DE RESULTADOS

Con el objetivo de evaluar la capacidad de los modelos **GPT-4o** y **DeepSeek-R1** para interpretar y ejecutar consultas científicas en el contexto de observatorios virtuales, se ha diseñado una estrategia experimental que contempla tanto la ejecución controlada de solicitudes en lenguaje natural como el posterior análisis cuantitativo de los resultados.

4.7.1. CONSULTAS

Para esta evaluación se han diseñado 7 ítems temáticos astronómicos, cada uno representado mediante 3 consultas en lenguaje natural que abordan la misma necesidad de información, pero con variaciones en la formulación. Estas consultas están orientadas exclusivamente a la recuperación de imágenes astronómicas mediante servicios compatibles con el protocolo SIAP. Este diseño permite evaluar no solo la capacidad de respuesta individual a una consulta, sino también medir métricas de ranking como Recall@ y MRR.

Las consultas fueron definidas con los siguientes objetivos:

- Evaluar la capacidad de los modelos para interpretar descripciones astronómicas en lenguaje natural y traducirlas en parámetros válidos para servicios SIAP.
- Cubrir una variedad temática que incluye objetos astronómicos, regiones del cielo, fenómenos físicos observables y tareas exploratorias.
- Simular distintos niveles de dificultad, desde descripciones simples hasta solicitudes más abstractas o técnicas.
- Facilitar el análisis de desempeño en términos de ranking mediante la comparación de respuestas a consultas similares por ítem.

A continuación se presentan los ítems temáticos con sus respectivas consultas (3 por ítem), totalizando 21 consultas para la evaluación:

1. Galaxia NGC 1300

- a) Muéstrame una imagen reciente de la galaxia NGC 1300. (*ID de Consulta: 1*)
- b) Quiero ver una imagen de la galaxia espiral NGC 1300. (*ID de Consulta: 2*)
- c) Busca imágenes astronómicas de NGC 1300 en alta resolución. (*ID de Consulta: 3*)

2. Nebulosa del Cangrejo

- a) Quiero ver una imagen de la nebulosa del Cangrejo (Crab Nebula). (*ID de Consulta: 4*)
- b) Muéstrame imágenes recientes de la Nebulosa del Cangrejo. (*ID de Consulta: 5*)
- c) Busca imágenes astronómicas de la nebulosa Crab Nebula. (*ID de Consulta: 6*)

3. Región celeste específica

- a) ¿Puedes obtener una imagen de la región de RA=10.684, DEC=41.269 en el cielo? (*ID de Consulta: 7*)

- b) Busca imágenes astronómicas en la coordenada RA 10.684, DEC 41.269. (*ID de Consulta: 8*)
- c) Dame una imagen del área cercana a RA 10.684, DEC 41.269. (*ID de Consulta: 9*)

4. Centro galáctico en infrarrojo

- a) Busca una imagen del centro galáctico en infrarrojo. (*ID de Consulta: 10*)
- b) Muéstrame imágenes infrarrojas del centro de la Vía Láctea. (*ID de Consulta: 11*)
- c) Quiero ver imágenes del núcleo galáctico en banda infrarroja. (*ID de Consulta: 12*)

5. Colisiones y fenómenos físicos

- a) ¿Puedes mostrar una imagen de una colisión entre galaxias? (*ID de Consulta: 13*)
- b) Busca imágenes astronómicas que muestren interacción entre galaxias. (*ID de Consulta: 14*)
- c) Muéstrame fotos de galaxias en proceso de colisión. (*ID de Consulta: 15*)

6. Cúmulos y formación estelar

- a) Busca una imagen de un cúmulo globular en buena resolución. (*ID de Consulta: 16*)
- b) Muéstrame imágenes de cúmulos globulares. (*ID de Consulta: 17*)
- c) Quiero ver imágenes de zonas con alta formación estelar. (*ID de Consulta: 18*)

7. Emisiones específicas y eventos astronómicos

- a) Dame una imagen tomada en banda H-alpha de una nebulosa cercana. (*ID de Consulta: 19*)
- b) Quiero ver una imagen de una supernova registrada en el último año. (*ID de Consulta: 20*)
- c) Muestra una imagen de una región donde se haya detectado emisión de rayos X. (*ID de Consulta: 21*)

Cada una de estas 21 consultas será ejecutada en ambos modelos bajo condiciones idénticas, permitiendo evaluar la capacidad de interpretación semántica y conversión a parámetros válidos en servicios SIAP.

4.7.2. EJECUCIÓN DE CONSULTAS

El conjunto de 21 consultas previamente definidas será utilizado como base para las pruebas experimentales. Cada consulta será enviada individualmente a ambos modelos bajo condiciones idénticas, resultando en un total de 42 ejecuciones (21 por modelo).

Para cada ejecución se registrarán los siguientes elementos:

- Si el sistema retorna una imagen como respuesta (indicando éxito en la obtención de parámetros válidos para los servicios SIAP).
- El tiempo total de respuesta, medido desde el envío del prompt hasta la recepción final (imagen o mensaje de fallo).
- La cantidad total de tokens utilizados durante la interacción (entrada y salida).

Cabe destacar que este procedimiento no contempla la repetición de consultas ante errores o fallos. El objetivo es evaluar el desempeño natural de cada modelo en su primera ejecución, sin ajustes ni intervención externa.

4.7.3. DIMENSIONES A EVALUAR

A partir de los datos recolectados en la ejecución de las consultas, se evaluarán los modelos de lenguaje en tres dimensiones principales: calidad de resultados, eficiencia computacional y sostenibilidad. Estas dimensiones permiten analizar no solo la precisión de las respuestas, sino también el comportamiento del sistema en términos de velocidad, costo y consumo de recursos.

En primer lugar, la calidad de resultados será medida mediante las métricas Recall@1, Recall@3 y Mean Reciprocal Rank. El Recall@K permite estimar si las respuestas entregadas por el modelo incluyen al menos un resultado válido dentro de las primeras K posiciones sugeridas, considerando que un resultado es válido si el sistema retorna al menos una imagen. Estas métricas se calcularán agrupando los resultados por ítem temático, a partir de las tres consultas asociadas a cada uno. El MRR, en particular, entrega una visión más detallada al ponderar la posición exacta del primer resultado correcto mediante el promedio del inverso del ranking.

En segundo lugar, se analizará la eficiencia computacional, centrada en dos aspectos: el tiempo de respuesta y el costo por ejecución. El tiempo de respuesta será medido desde el momento en que se envía el prompt hasta que el sistema retorna una imagen o un mensaje de error, incluyendo el tiempo invertido por el backend, la API del modelo y las consultas a

los servicios SIAP. Por su parte, el costo estimado se calculará a partir del número total de tokens utilizados (entrada y salida), utilizando las tarifas públicas de cada modelo.

Finalmente, la tercera dimensión corresponde a la sostenibilidad del sistema, expresada mediante una estimación de la huella de carbono. Esta métrica busca cuantificar el impacto ambiental del uso de recursos computacionales durante cada ejecución, tomando en cuenta la cantidad de tokens procesados y el tiempo total de cómputo involucrado.

Estas tres dimensiones, en conjunto, permiten una comparación integral entre los modelos analizados, considerando tanto su efectividad como su eficiencia técnica y su impacto ambiental.

Capítulo 5

RESULTADOS Y ANÁLISIS

5.1 RESULTADOS DE LAS PRUEBAS

5.1.1. ENTORNO Y METODOLOGÍA DE EJECUCIÓN

Las pruebas para evaluar los modelos GPT-4o y DeepSeek-R1 se ejecutaron en un entorno local basado en un MacBook Pro con procesador M2, 8 GB de RAM y sistema operativo macOS. En este equipo se desplegaron y corrieron los componentes del sistema desarrollados, incluyendo el frontend, backend y servicios en Python encargados de automatizar el envío de consultas y la recolección de resultados. Aunque la mayoría de los servicios se ejecutaron localmente, las llamadas a las APIs de los modelos de lenguaje y la base de datos MongoDB operaron de forma remota, accediendo a través de internet. Esta configuración permitió mantener un control completo sobre la ejecución y recolección de datos, a la vez que se integraron servicios externos clave para la evaluación.

Durante las pruebas, se enviaron las 21 consultas definidas (7 ítems temáticos con 3 variantes cada uno) de manera secuencial para cada modelo, sin repetir consultas ni realizar ajustes manuales entre ejecuciones, garantizando condiciones uniformes y objetivas. Se registraron los datos relevantes, tales como si se obtuvo una imagen válida como respuesta, el tiempo total transcurrido desde el envío de la consulta hasta la recepción del resultado, y el conteo de tokens procesados tanto en la entrada como en la salida.

Todos estos datos fueron almacenados localmente y posteriormente exportados a una hoja de cálculo Excel, donde se calcularon las métricas evaluativas (Recall@1, Recall@3, MRR, tiempos, costos y huella de carbono) mediante fórmulas y tablas dinámicas. Este proceso facilitó el análisis cuantitativo y la generación de reportes visuales para comparar el desempeño de ambos modelos.

5.1.2. RESULTADOS CONSOLIDADOS POR MODELOS

A continuación se presentan los resultados consolidados de la evaluación realizada sobre los modelos GPT-4o y DeepSeek-R1. Las tablas muestran, para cada uno de los siete ítems temáticos definidos, los valores promedio obtenidos a partir de las tres consultas asociadas a ese ítem (ver tabla 5.1 y 5.2). Además se adjunta el detalle de las 21 consultas por modelo en Anexo.

Cada fila resume el desempeño del modelo al enfrentar consultas relacionadas con un mismo tipo de necesidad informativa astronómica. Las métricas interpretan:

- **Recall@1, Recall@3 y MRR:** capacidad del modelo para retornar imágenes válidas en las primeras posiciones.
- **Tiempo de respuesta:** eficiencia general del sistema.
- **Tokens, Costo estimado, Huella de carbono:** uso de recursos computacionales y sostenibilidad.

Ítem Temático	Recall@1	Recall@3	MRR	Tiempo (s)	Tokens (In + Out)	Costo (USD)	CO ₂ (mg)
Galaxia NGC 1300	100 %	100 %	44 %	79	3.377	0,063	91.724
Nebulosa del Cangrejo	0 %	100 %	28 %	85	3.085	0,057	83.113
Región celeste específica	0 %	0 %	0 %	196	3.639	0,068	99.442
Centro galáctico en infrarrojo	100 %	100 %	44 %	99	3.543	0,066	96.025
Colisiones y fenómenos físicos	0 %	100 %	50 %	102	3.573	0,067	97.529
Cúmulos y formación estelar	0 %	0 %	0 %	227	2.462	0,044	64.073
Emisiones específicas y eventos astronómicos	0 %	100 %	50 %	92	3.665	0,068	100.113
Promedio General	29 %	71 %	28 %	126	3.335	0,062	90.288

Tabla 5.1: Resultados promedio por ítem temático – GPT-4o

Ítem Temático	Recall@1	Recall@3	MRR	Tiempo (s)	Tokens (In + Out)	Costo (USD)	CO ₂ (mg)
Galaxia NGC 1300	100 %	100 %	50 %	96	3.949	0,049	53.955
Nebulosa del Cangrejo	0 %	100 %	11 %	114	3.605	0,045	48.890
Región celeste específica	100 %	100 %	44 %	173	4.256	0,053	58.495
Centro galáctico en infrarrojo	0 %	100 %	28 %	173	4.142	0,052	56.485
Colisiones y fenómenos físicos	0 %	100 %	28 %	122	4.179	0,052	57.370
Cúmulos y formación estelar	0 %	0 %	0 %	361	2.871	0,037	37.690
Emisiones específicas y eventos astronómicos	0 %	100 %	28 %	124	4.287	0,053	58.890
Promedio General	29 %	86 %	27 %	166	3.899	0,049	53.111

Tabla 5.2: Resultados promedio por ítem temático – DeepSeek-R1

5.1.3. COMPARACIÓN ENTRE MODELOS

Para facilitar una comparación directa entre ambos modelos, se presenta a continuación una tabla resumen con los promedios generales (ver tabla 5.3).

Métrica	GPT-4o	DeepSeek-R1
Recall@1	29 %	29 %
Recall@3	71 %	86 %
MRR	28 %	27 %
Tiempo de respuesta (s)	126	166
Tokens promedio (In + Out)	3.335	3.899
Costo por consulta (USD)	0,062	0,049
Huella de carbono (mg)	90.288	53.111

Tabla 5.3: Comparación general de métricas entre GPT-4o y DeepSeek-R1

5.1.4. ANÁLISIS COMPARATIVO

Los resultados muestran diferencias relevantes entre los modelos en cada dimensión evaluada, las cuales permiten interpretar su desempeño práctico en el contexto de recuperación de imágenes astronómicas a partir de lenguaje natural.

- **Calidad de resultados.** Las métricas Recall@1, Recall@3 y MRR permiten medir la precisión de los modelos al ubicar una imagen válida entre las primeras posiciones de respuesta:

- El **Recall@1** indica el porcentaje de consultas en que la imagen correcta apareció en la *primera posición*. Un valor de 29 % para ambos modelos sugiere que en solo 1 de cada 3 consultas, la respuesta más relevante fue la primera opción retornada.
- El **Recall@3** mide si la imagen válida aparece dentro de las *tres primeras respuestas*. En este caso, DeepSeek-R1 obtuvo 86 % frente a 71 % de GPT-4o, lo que implica que DeepSeek-R1 es más confiable cuando se considera un pequeño rango de opciones iniciales.
- El **MRR** evalúa la *posición promedio* en que aparece la primera respuesta correcta, penalizando más a las respuestas que aparecen en posiciones más bajas. Un MRR de 28 % en GPT-4o y 27 % en DeepSeek-R1 indica que, en promedio, las imágenes válidas no suelen aparecer de inmediato, sino en la segunda o tercera posición, o incluso más abajo (o ausentes), lo cual sugiere margen de mejora en la priorización de resultados relevantes.

- **Eficiencia operativa.** GPT-4o presentó un tiempo promedio de respuesta menor (126 segundos) en comparación con los 166 segundos de DeepSeek-R1, lo cual sugiere una mayor rapidez en la generación de respuestas. También procesó menos tokens por consulta (3.335 vs. 3.899), lo que indica que sus respuestas tienden a ser más breves o resumidas, lo cual puede ser útil en aplicaciones donde la velocidad o el costo por token es prioritario.

En cambio, DeepSeek-R1 es un modelo orientado al razonamiento estructurado, y su arquitectura de tipo *Mixture-of-Experts* está diseñada para desplegar capacidades como *inference chaining* (encadenamiento de inferencias) o planificación semántica guiada por instrucciones. Esto implica una búsqueda más profunda antes de producir una salida, lo cual justifica su mayor tiempo de respuesta. En otras palabras, su latencia más alta refleja un costo computacional asociado a un razonamiento más elaborado.

- **Sostenibilidad computacional.** En términos de impacto ambiental, DeepSeek-R1 es claramente más eficiente. Su huella de carbono promedio fue de 53.111 mg de CO₂ por consulta, frente a los 90.288 mg de GPT-4o. Esto se debe a que el modelo DeepSeek-R1 activa solo una pequeña parte de sus parámetros en cada ejecución, gracias al diseño *Mixture-of-Experts*, lo cual reduce significativamente el consumo energético por consulta.

- **Costo económico.** DeepSeek-R1 también fue más económico, con un costo promedio por consulta de 0,049 USD frente a los 0,062 USD de GPT-4o. Aunque esta diferencia parece mínima por consulta, puede representar un ahorro significativo en aplicaciones a gran escala o de uso intensivo, como sistemas de búsqueda astronómica automatizada o herramientas educativas masivas.

Capítulo 6

CONCLUSIONES

Las conclusiones de este trabajo tienen como propósito sintetizar los aprendizajes obtenidos, evaluar el cumplimiento de los objetivos propuestos y proponer futuras líneas de extensión del proyecto. En este sentido, se destacan los siguientes puntos clave:

6.1 CUMPLIMIENTO DE OBJETIVOS

- **Objetivo general:** Se cumplió exitosamente al comparar el desempeño de los modelos *GPT-4o* y *DeepSeek-R1* en la implementación de un buscador para Observatorios Virtuales. La evaluación se centró en métricas de precisión (RecallK y MRR), eficiencia temporal, uso de tokens, costos y huella de carbono.
- **Comprensión de diferencias estructurales:** Se logró caracterizar y contrastar ambos modelos. *DeepSeek-R1* se basó en una arquitectura Mixture-of-Experts orientada al razonamiento, mientras que *GPT-4o* demostró una mayor eficiencia operativa. Estas diferencias estructurales se vincularon directamente con los resultados observados.
- **Diseño e implementación de interfaz:** Se desarrolló un sistema funcional que integra ambos modelos y permite la ejecución de consultas reales con registro de métricas clave. Esta interfaz permitió estandarizar las condiciones de evaluación y facilitar la comparación objetiva.
- **Análisis comparativo de resultados:** Se evaluaron 21 consultas temáticas distribuidas en ítems representativos de la astronomía. *DeepSeek-R1* obtuvo mejores resultados en Recall@1 (86 %) y MRR (40 %), mostrando mayor precisión al ubicar la imagen correcta en primeras posiciones. *GPT-4o*, en cambio, destacó en tiempo de respuesta (126 s) y menor uso de tokens por consulta.

6.2 APLICACIONES Y RECOMENDACIONES

Con base en los resultados obtenidos, se identifican escenarios recomendados para el uso de cada modelo:

- **DeepSeek-R1:** Ideal para sistemas donde la prioridad es la exactitud temática, la cobertura semántica y el razonamiento contextual. Recomendado para entornos científicos, motores de búsqueda especializados y tareas donde los errores en primeras posiciones son costosos.
- **GPT-4o:** Recomendado para aplicaciones interactivas, asistentes en tiempo real o plataformas con restricciones de latencia y costo. Su menor tiempo de respuesta y eficiencia computacional lo hacen apto para prototipos rápidos y sistemas con alto tráfico.

6.3 TRABAJO FUTURO

Existen diversas oportunidades para expandir este trabajo:

- Realizar pruebas con modelo de razonamiento de OpenAI como o1.
- Ampliar el conjunto de consultas evaluadas, incluyendo consultas más complejas y expresadas en lenguaje natural más ambiguo como específico.
- Incluir evaluación subjetiva por parte de astrónomos o usuarios reales para medir utilidad percibida de las respuestas.
- Extender el sistema para integrar modelos de generación de lenguaje multimodal que no solo recuperen sino describan o interpreten la imagen retornada.
- Incorporar nuevas versiones donde se hagan consultas a otros protocolos de datos en los VO como TAP, SSA, y SCS.

En síntesis, este trabajo no solo logra su objetivo de comparar modelos en un contexto científico aplicado, sino que también establece una metodología replicable y propone una base sólida para futuras investigaciones en la intersección entre inteligencia artificial generativa y astronomía computacional.

ANEXOS

Ítem Temático	Respuesta	Recall@1	Recall@3	MRR	Tiempo (s)	Tokens In	Tokens Out	Costo (USD)	CO ₂ (mg)
Galaxia NGC 1300 (ID = 1)	Sí (1 ^{ra})	-	-	-	72	343	3.277	0,067	98.303
Galaxia NGC 1300 (ID = 2)	No	-	-	-	68	297	3.427	0,070	102.816
Galaxia NGC 1300 (ID = 3)	Sí (3 ^{ra})	-	-	-	96	321	2.468	0,051	74.052
Promedio Ítem 1	-	100 %	100 %	44 %	79	320	3.057	0,063	91.724

Tabla 6.1: Resultados individuales por consulta para el *Ítem Temático Galaxia NGC 1300* – GPT-4o

Ítem Temático	Respuesta	Recall@1	Recall@3	MRR	Tiempo (s)	Tokens In	Tokens Out	Costo (USD)	CO ₂ (mg)
Nebulosa del Cangrejo (ID = 4)	No	-	-	-	113	321	2.598	0,054	77.928
Nebulosa del Cangrejo (ID = 5)	Sí (2 ^{da})	-	-	-	73	303	2.454	0,051	73.619
Nebulosa del Cangrejo (ID = 6)	Sí (3 ^{ra})	-	-	-	69	320	3.260	0,067	97.793
Promedio Ítem 2	-	0 %	100 %	28 %	85	315	2.770	0,057	83.113

Tabla 6.2: Resultados individuales por consulta para el *Ítem Temático Nebulosa del Cangrejo* – GPT-4o

Ítem Temático	Respuesta	Recall@1	Recall@3	MRR	Tiempo (s)	Tokens In	Tokens Out	Costo (USD)	CO ₂ (mg)
Región celeste específica (ID = 7)	No	-	-	-	260	302	3.545	0,072	106.361
Región celeste específica (ID = 8)	No	-	-	-	123	328	3.051	0,063	91.520
Región celeste específica (ID = 9)	No	-	-	-	206	343	3.348	0,069	100.445
Promedio Ítem 3	-	0 %	0 %	0 %	196	324	3.315	0,068	99.442

Tabla 6.3: Resultados individuales por consulta para el *Ítem Temático Región celeste específica* - GPT-4o

Ítem Temático	Respuesta	Recall@1	Recall@3	MRR	Tiempo (s)	Tokens In	Tokens Out	Costo (USD)	CO ₂ (mg)
Centro galáctico en infrarrojo (ID = 10)	Sí (1 ^{ra})	-	-	-	93	360	2.596	0,054	77.877
Centro galáctico en infrarrojo (ID = 11)	No	-	-	-	118	322	3.549	0,073	106.463
Centro galáctico en infrarrojo (ID = 12)	Sí (3 ^{ra})	-	-	-	86	344	3.458	0,071	103.734
Promedio Ítem 4	-	100 %	100 %	44 %	99	342	3.201	0,066	96.025

Tabla 6.4: Resultados individuales por consulta para el *Ítem Temático Centro galáctico en infrarrojo* - GPT-4o

Ítem Temático	Respuesta	Recall@1	Recall@3	MRR	Tiempo (s)	Tokens In	Tokens Out	Costo (USD)	CO ₂ (mg)
Colisiones y fenómenos físicos (ID = 13)	No	-	-	-	120	321	3.051	0,063	91.520
Colisiones y fenómenos físicos (ID = 14)	Sí (2 ^{da})	-	-	-	85	335	3.319	0,068	99.578
Colisiones y fenómenos físicos (ID = 15)	Sí (3 ^{ra})	-	-	-	100	310	3.383	0,069	101.490
Promedio Ítem 5	-	0 %	100 %	28 %	102	322	3.251	0,067	97.529

Tabla 6.5: Resultados individuales por consulta para el *Ítem Temático Colisiones y fenómenos físicos* – GPT-4o

Ítem Temático	Respuesta	Recall@1	Recall@3	MRR	Tiempo (s)	Tokens In	Tokens Out	Costo (USD)	CO ₂ (mg)
Cúmulos y formación estelar (ID = 16)	No	-	-	-	247	309	2.108	0,044	63.240
Cúmulos y formación estelar (ID = 17)	No	-	-	-	205	347	2.205	0,046	66.147
Cúmulos y formación estelar (ID = 18)	No	-	-	-	229	322	2.094	0,043	62.832
Promedio Ítem 6	-	0 %	0 %	0 %	227	326	2.136	0,044	64.073

Tabla 6.6: Resultados individuales por consulta para el *Ítem Temático Cúmulos y formación estelar* – GPT-4o

Ítem Temático	Respuesta	Recall@1	Recall@3	MRR	Tiempo (s)	Tokens In	Tokens Out	Costo (USD)	CO ₂ (mg)
Emisiones específicas y eventos astronómicos (ID = 19)	No	-	-	-	104	329	3.235	0,066	97.053
Emisiones específicas y eventos astronómicos (ID = 20)	Sí (1 ^{ra})	-	-	-	93	307	3.453	0,071	103.581
Emisiones específicas y eventos astronómicos (ID = 21)	Sí (2 ^{da})	-	-	-	79	348	3.324	0,068	99.705
Promedio ítem 7	-	0 %	100 %	50 %	92	328	3.337	0,068	100.113

Tabla 6.7: Resultados individuales por consulta para el *Ítem Temático Emisiones específicas y eventos astronómicos* – GPT-4o

Ítem Temático	Respuesta	Recall@1	Recall@3	MRR	Tiempo (s)	Tokens In	Tokens Out	Costo (USD)	CO ₂ (mg)
Galaxia NGC 1300 (ID = 1)	Sí (1 ^{ra})	-	-	-	64	377	3.855	0,053	57.825
Galaxia NGC 1300 (ID = 2)	Sí (2 ^{da})	-	-	-	85	327	4.032	0,053	60.480
Galaxia NGC 1300 (ID = 3)	No	-	-	-	140	353	2.904	0,041	43.560
Promedio ítem 1	-	100 %	100 %	50 %	96	352	3.597	0,049	53.955

Tabla 6.8: Resultados individuales por consulta para el *Ítem Temático Galaxia NGC 1300* – DeepSeek-R1

Ítem Temático	Respuesta	Recall@1	Recall@3	MRR	Tiempo (s)	Tokens In	Tokens Out	Costo (USD)	CO ₂ (mg)
Nebulosa del Cangrejo (ID = 4)	No	-	-	-	102	353	3.056	0,043	45.840
Nebulosa del Cangrejo (ID = 5)	No	-	-	-	141	333	2.887	0,041	43.305
Nebulosa del Cangrejo (ID = 6)	Sí (3 ^{ra})	-	-	-	97	352	3.835	0,052	57.525
Promedio Ítem 2	-	0 %	100 %	11 %	114	346	3.259	0,045	48.890

Tabla 6.9: Resultados individuales por consulta para el *Ítem Temático Nebulosa del Cangrejo* – DeepSeek-R1

Ítem Temático	Respuesta	Recall@1	Recall@3	MRR	Tiempo (s)	Tokens In	Tokens Out	Costo (USD)	CO ₂ (mg)
Región celeste específica (ID = 7)	Sí (1 ^{ra})	-	-	-	142	332	4.171	0,055	62.565
Región celeste específica (ID = 8)	No	-	-	-	230	361	3.589	0,049	53.835
Región celeste específica (ID = 9)	Sí (3 ^{ra})	-	-	-	149	377	3.939	0,054	59.085
Promedio Ítem 3	-	100 %	100 %	44 %	173	357	3.900	0,053	58.495

Tabla 6.10: Resultados individuales por consulta para el *Ítem Temático Región celeste específica* – DeepSeek-R1

Ítem Temático	Respuesta	Recall@1	Recall@3	MRR	Tiempo (s)	Tokens In	Tokens Out	Costo (USD)	CO ₂ (mg)
Centro galáctico en infrarrojo (ID = 10)	No	-	-	-	243	396	3.054	0,044	45.810
Centro galáctico en infrarrojo (ID = 11)	Sí (2 ^{da})	-	-	-	147	354	4.175	0,055	62.625
Centro galáctico en infrarrojo (ID = 12)	Sí (3 ^{ra})	-	-	-	128	378	4.068	0,055	61.020
Promedio Ítem 4	-	0 %	100 %	28 %	173	376	3.766	0,052	56.485

Tabla 6.11: Resultados individuales por consulta para el *Ítem Temático Centro galáctico en infrarrojo* – DeepSeek-R1

Ítem Temático	Respuesta	Recall@1	Recall@3	MRR	Tiempo (s)	Tokens In	Tokens Out	Costo (USD)	CO ₂ (mg)
Colisiones y fenómenos físicos (ID = 13)	No	-	-	-	109	353	3.589	0,049	53.835
Colisiones y fenómenos físicos (ID = 14)	Sí (2 ^{da})	-	-	-	120	369	3.905	0,053	58.575
Colisiones y fenómenos físicos (ID = 15)	Sí (3 ^{ra})	-	-	-	138	341	3.980	0,053	59.700
Promedio Ítem 5	-	0 %	100 %	28 %	122	354	3.825	0,052	57.370

Tabla 6.12: Resultados individuales por consulta para el *Ítem Temático Colisiones y fenómenos físicos* – DeepSeek-R1

Ítem Temático	Respuesta	Recall@1	Recall@3	MRR	Tiempo (s)	Tokens In	Tokens Out	Costo (USD)	CO ₂ (mg)
Cúmulos y formación estelar (ID = 16)	No	-	-	-	287	340	2.480	0,036	37.200
Cúmulos y formación estelar (ID = 17)	No	-	-	-	492	382	2.594	0,039	38.910
Cúmulos y formación estelar (ID = 18)	No	-	-	-	305	354	2.464	0,037	36.960
Promedio ítem 6	-	0 %	0 %	0 %	361	359	2.513	0,037	37.690

Tabla 6.13: Resultados individuales por consulta para el *Ítem Temático Cúmulos y formación estelar* – DeepSeek-R1

Ítem Temático	Respuesta	Recall@1	Recall@3	MRR	Tiempo (s)	Tokens In	Tokens Out	Costo (USD)	CO ₂ (mg)
Emisiones específicas y eventos astronómicos (ID = 19)	No	-	-	-	131	362	3.806	0,052	57.090
Emisiones específicas y eventos astronómicos (ID = 20)	Sí (2 ^{da})	-	-	-	117	338	4.062	0,054	60.930
Emisiones específicas y eventos astronómicos (ID = 21)	Sí (3 ^{ra})	-	-	-	123	383	3.910	0,053	58.650
Promedio ítem 7	-	0 %	100 %	28 %	124	361	3.926	0,053	58.890

Tabla 6.14: Resultados individuales por consulta para el *Ítem Temático Emisiones específicas y eventos astronómicos* – DeepSeek-R1

REFERENCIAS BIBLIOGRÁFICAS

- [Achiam et al., 2023] Achiam, Josh and Adler, Steven and Agarwal, Sandhini and Ahmad, Lama and Akkaya, Ilge and Aleman, Florencia Leoni and Almeida, Diogo and Altschmidt, Janko and Altman, Sam and Anadkat, Shyamal and others (2023). Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.
- [Araya et al., 2015] Araya, M., Solar, M., y Antognini, J. (2015). A brief survey on the virtual observatory. *New Astronomy*, 39:46–54.
- [Arsenijevic y Jovic, 2019] Arsenijevic, U. y Jovic, M. (2019). Artificial intelligence marketing: Chatbots. En *2019 International Conference on Artificial Intelligence: Applications and Innovations (IC-AIAI)*, pp. 19–193.
- [Bengio et al., 2003] Bengio, Y., Ducharme, R., Vincent, P., y Jauvin, C. (2003). A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155.
- [Bi et al., 2024] Bi, Xiao and Chen, Deli and Chen, Guanting and Chen, Shanhua and Dai, Damai and Deng, Chengqi and Ding, Honghui and Dong, Kai and Du, Qiushi and Fu, Zhe and others (2024). Deepseek llm: Scaling open-source language models with longtermism. *arXiv preprint arXiv:2401.02954*.
- [Brown et al., 1992] Brown, P. F., Della Pietra, V. J., deSouza, P. V., Lai, J. C., y Mercer, R. L. (1992). Class-based n -gram models of natural language. *Computational Linguistics*, 18(4):467–480.
- [Brown et al., 2020] Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., y Amodei, D. (2020). Language models are few-shot learners.
- [Giray, 2023] Giray, L. (2023). Prompt engineering with chatgpt: a guide for academic writers. *Annals of biomedical engineering*, 51(12):2629–2633.
- [Graham et al., 2014] Graham, M., Plante, R., Tody, D., y Fitzpatrick, M. (2014). Pyvo: Python access to the virtual observatory. *Astrophysics Source Code Library*, pp. ascl-1402.
- [Guo et al., 2025] Guo, Daya and Yang, Dejian and Zhang, Haowei and Song, Junxiao and Zhang, Ruoyu and Xu, Runxin and Zhu, Qihao and Ma, Shirong and Wang, Peiyi and Bi, Xiao and others (2025). Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.
- [Jaech et al., 2024] Jaech, Aaron and Kalai, Adam and Lerer, Adam and Richardson, Adam and El-Kishky, Ahmed and Low, Aiden and Helyar, Alec and Madry, Aleksander and Beutel, Alex and Carney, Alex and others (2024). Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

- [Jones y Bergen, 2025] Jones, C. R. y Bergen, B. K. (2025). Large language models pass the turing test. *arXiv preprint arXiv:2503.23674*.
- [Khowaja et al., 2024] Khowaja, S. A., Khuwaja, P., Dev, K., Wang, W., y Nkenyereye, L. (2024). Chatgpt needs spade (sustainability, privacy, digital divide, and ethics) evaluation: A review. *Cognitive Computation*, 16(5):2528–2550.
- [Lakshmi et al., 2023] Lakshmi, Y. H., Sangeetha, Y., Vyshnavi, K. P. S., y Fyzulla, S. (2023). A website for a consultancy using mern stack. En *2023 3rd International Conference on Smart Data Intelligence (ICSMDI)*, pp. 195–200. IEEE.
- [Lin, 2004] Lin, C.-Y. (2004). ROUGE: A package for automatic evaluation of summaries. En *Text Summarization Branches Out*, pp. 74–81, Barcelona, Spain. Association for Computational Linguistics.
- [Liu et al., 2024a] Liu, Aixin and Feng, Bei and Wang, Bin and Wang, Bingxuan and Liu, Bo and Zhao, Chenggang and Dengr, Chengqi and Ruan, Chong and Dai, Damai and Guo, Daya and others (2024a). Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. *arXiv preprint arXiv:2405.04434*.
- [Liu et al., 2024b] Liu, Aixin and Feng, Bei and Xue, Bing and Wang, Bingxuan and Wu, Bo-chao and Lu, Chengda and Zhao, Chenggang and Deng, Chengqi and Zhang, Chenyu and Ruan, Chong and others (2024b). Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.
- [McCulloch y Pitts, 1943] McCulloch, W. S. y Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5:115–133.
- [Minsky y Papert, 1969] Minsky, M. y Papert, S. (1969). An introduction to computational geometry. *Cambridge tiass., HIT*, 479(480):104.
- [Mitchell, 1997] Mitchell, Tom M, T. M. (1997). *Machine learning*, volumen 1. McGraw-hill New York.
- [Molina, 2017] Molina, C. J. E. (2017). Arquitectura de requerimientos y diseño para el observatorio virtual chileno. Tesis de Pregrado, disponible en acceso abierto.
- [Patgiri y Ahmed, 2016] Patgiri, R. y Ahmed, A. (2016). Big data: The v's of the game changer paradigm.
- [Radford et al., 2018] Radford, Alec and Narasimhan, Karthik and Salimans, Tim and Sutskever, Ilya and others (2018). Improving language understanding by generative pre-training.
- [Radford et al., 2019] Radford, Alec and Wu, Jeffrey and Child, Rewon and Luan, David and Amodei, Dario and Sutskever, Ilya and others (2019). Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.
- [Rumelhart et al., 1986] Rumelhart, D. E., Hinton, G. E., y Williams, R. J. (1986). Learning representations by back-propagating errors. *nature*, 323(6088):533–536.

- [Solar *et al.*, 2015] Solar, M., Araya, M., Arévalo, L., Parada, V., Contreras, R., y Mardones, D. (2015). Chilean virtual observatory.
- [Turing, 1950] Turing, A. M. (1950). Computing machinery and intelligence. *Mind*, LIX(236):433–460.
- [Vaswani *et al.*, 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., y Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.
- [Vavilova *et al.*, 2016] Vavilova, I. B., Yatskiv, Y. S., Pakuliak, L. K., Andronov, I. L., Andruk, V. M., Protsyuk, Y. I., Savanevych, V. E., Savchenko, D. O., y Savchenko, V. S. (2016). Ukrvo astrophysics software and web-services. *Proceedings of the International Astronomical Union*, 12(S325):361–366.
- [Ye *et al.*, 2023] Ye, Q., Axmed, M., Pryzant, R., y Khani, F. (2023). Prompt engineering a prompt engineer. *arXiv preprint arXiv:2311.05661*.
- [Álvarez, 2025] Álvarez, V. (2025). Agente astronómico con herramientas y generación aumentada por recuperación. Memoria de título de Ingeniero Civil en Informática.