**UNIVERSIDAD TÉCNICA DEL NORTE**

**FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS**

**CARRERA DE SOFTWARE**

**TEMA:**

# DESARROLLO DE UN ASISTENTE VIRTUAL ACADEMICÓ BASADO EN UN MODELO DE LENGUAJE PREENTRENADO(LLM) PARA ESTUDIANTES DE LA FICA

Trabajo de integración curricular previo a la obtención del título de ingeniero de Software


**AUTOR:**

Dylan Jesús Ortega Freire

**DIRECTOR:**

PhD. Iván Danilo García Santillán


Ibarra – Ecuador

**2026**

# UNIVERSIDAD TÉCNICA DEL NORTE
## BIBLIOTECA UNIVERSITARIA

### AUTORIZACIÓN DE USO Y PUBLICACIÓN A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

**1. IDENTIFICACIÓN DE LA OBRA**

En cumplimiento del Art. 144 de la Ley de Educación Superior, hago la entrega del presente trabajo a la Universidad Técnica del Norte para que sea publicado en el Repositorio Digital Institucional, para lo cual pongo a disposición la siguiente información:

| DATOS DE CONTACTO | | | |
|---|---|---|---|
| **CÉDULA DE IDENTIDAD:** | 080278111-2 | | |
| **APELLIDOS Y NOMBRES:** | ORTEGA FREIRE DYLAN JESUS | | |
| **DIRECCIÓN:** | ARSENIO TORRES 2-37 | | |
| **EMAIL:** | djortegaf@utn.edu.ec / dylanjesus65@gmail.com | | |
| **TELÉFONO FIJO:** | | **TELÉFONO MÓVIL:** | 0999379876 |

| DATOS DE LA OBRA | |
|---|---|
| **TÍTULO:** | DESARROLLO DE UN ASISTENTE VIRTUAL ACADEMICÓ BASADO EN UN MODELO DE LENGUAJE PREENTRENADO(LLM) PARA ESTUDIANTES DE LA FICA |
| **AUTOR (ES):** | ORTEGA FREIRE DYLAN JESUS |
| **FECHA DE APROBACIÓN: DD/MM/AAAA** | 04/02/2026 |
| **PROGRAMA:** | ■ **PREGRADO** ☐ **POSGRADO** |
| **TITULO POR EL QUE OPTA:** | INGENIERIO EN SOFTWARE |
| **ASESOR /DIRECTOR:** | PhD. GARCÍA SANTILLLÁN IVÁN DANILO |

# UNIVERSIDAD TÉCNICA DEL NORTE
## BIBLIOTECA UNIVERSITARIA

### AUTORIZACIÓN DE USO Y PUBLICACIÓN A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

**2. CONSTANCIAS**

El autor manifiesta que la obra objeto de la presente autorización es original y se la desarrolló, sin violar derechos de autor de terceros, por lo tanto, la obra es original y que es el titular de los derechos patrimoniales, por lo que asume la responsabilidad sobre el contenido de la misma y saldrá en defensa de la Universidad en caso de reclamación por parte de terceros.

Ibarra, a los 4 días, del mes de febrero de 2026

**EL AUTOR:**

DYLAN JESUS ORTEGA FREIRE

Firmado digitalmente por DYLAN JESUS ORTEGA FREIRE
Fecha: 2026.02.04 03:15:06 -05'00'

.....

Ortega Freire Dylan Jesús
CI: 0802781112

## CERTFICACIÓN DEL DIRECTOR

Ibarra, 04 de Febrero de 2026

## CERTIFICACIÓN DEL DIRECTOR DEL TRABAJO DE INTEGRACIÓN CURRICULAR

Por medio del presente, yo **PhD. Iván Danilo García Santillán**, certifico que el Sr. **Ortega Freire Dylan Jesús**, portador de la cédula de ciudadanía número **0802781112**, ha trabajado en el desarrollo del proyecto de grado titulado "**Desarrollo de un asistente virtual académico basado en un modelo de lenguaje preentrenado (LLM) para estudiantes de la FICA**", previo a la obtención del Título de Ingeniero en Software. Este trabajo se ha realizado con responsabilidad, lo cual certifico con honor a la verdad.

Atentamente:

IVAN DANILO GARCIA SANTILLAN

Digitally signed by IVAN DANILO GARCIA SANTILLAN
Date: 2026.02.04 09:06:03 -05'00'

.....

PhD. Iván Danilo García Santillán
C.C.: 1002292603
DIRECTOR DEL TRABAJO DE GRADO

## DEDICATORIA

A mis padres, Stela y Edy, por su apoyo constante e incondicional. Gracias a ellos he formado los valores que hoy me definen, y saber que se sienten orgullosos de mí es una de las principales razones por las que me esfuerzo por alcanzar este objetivo.

A mis hermanos menores, Ainhoa y Aarón, para que nunca se rindan y tengan siempre presente que ellos también pueden hacerlo.

A mis Abuelitos María y Lizardo por ser un apoyo más y nunca dejarme solo a ellos también les dedico este logro.

A Graciela, porque sé que, desde donde se encuentre, este logro la hará sentir feliz.

- **Dylan Jesús Ortega Freire**

## AGRADECIMIENTO

Quiero agradecer a Dios por cuidarme en cada etapa de este camino y por darme la fortaleza necesaria para mantenerme firme, incluso en los momentos más difíciles.

A mi familia en especial a mis padres, Edy y Stela les expreso mi más profundo agradecimiento por su apoyo incondicional y por brindarme la oportunidad de estudiar. Gracias a su esfuerzo, dedicación y confianza, nunca me faltó nada. Este logro también es de ustedes.

A mi Nikol y su familia, por su apoyo y fe en mí, en especial a Nikol, cuyo cariño ha sido un impulso constante y una motivación para superarme y alcanzar este objetivo.

A mis grandes amigos Joan, Isaac, Gustavo, Edwin y Jairo, gracias por ser un motor más y por estar siempre pendientes de mí durante esta etapa. Sin ustedes, estoy seguro de que lograr este objetivo me habría costado mucho más.

Quiero agradecer a mi director de tesis, el ingeniero Iván García, por la paciencia y el acompañamiento durante todo este proceso, De igual manera, agradezco a mi asesor, MacArthur Ortega, por el tiempo brindado y la constante disposición para ayudarme

- **Dylan Jesús Ortega Freire**

# LISTA DE SIGLAS

**UTN:** Universidad Técnica del Norte

**ASGI:** Asynchronous Server Gateway Interface

**BERT:** Bidirectional Encoder Representations from Transformers

**BLEU:** Bilingual Evaluation Understudy

**CPU:** Central Processing Unit (Unidad Central de Procesamiento)

**CRISP-DM:** Cross-Industry Standard Process for Data Mining

**CUQ:** Chatbot Usability Questionnaire

**FICA:** Facultad de Ingeniería en Ciencias Aplicadas

**GQA:** Grouped Query Attention

**GPT:** Generative Pre-trained Transformer

**GPU:** Graphics Processing Unit (Unidad de Procesamiento Gráfico)

**IA:** Inteligencia Artificial

**LCS:** Longest Common Subsequence

**LLM:** Large Language Model (Modelo de Lenguaje de Gran Escala)

**LoRA:** Low-Rank Adaptation

**NLP:** Natural Language Processing (Procesamiento del Lenguaje Natural)

**OCR:** Optical Character Recognition (Reconocimiento Óptico de Caracteres)

**ODS:** Objetivos de Desarrollo Sostenible

**PEFT:** Parameter-Efficient Fine-Tuning

**PLN:** Procesamiento del Lenguaje Natural

**PPL:** Perplexity (Perplejidad)

**QA:** Question-Answering (Pregunta-Respuesta)

**QLoRA:** Quantized Low-Rank Adaptation

**RAG:** Retrieval-Augmented Generation (Generación Aumentada por Recuperación)

**UTN:** Universidad Técnica del Norte

## INDICE DE CONTENIDOS

**INDICE DE TABLAS**

**INDICE DE FIGURAS**

## RESUMEN

El acceso a la normativa de la Facultad de Ingeniería en Ciencias Aplicadas (FICA) de la Universidad Técnica del Norte suele ser un proceso poco eficiente, dado que estudiantes y docentes deben revisar manualmente documentos extensos, lo que dificulta obtener información específica de manera rápida y confiable. Para abordar esta cuestión, este trabajo propone la creación de un asistente virtual fundamentado en un modelo lingüístico de gran tamaño (LLM), al que se le reentrenará utilizando técnicas de fine-tuning, para así ofrecer respuestas precisas y contextualizadas a preguntas sobre la normativa institucional. El modelo CRISP-DM fue el método utilizado, lo que posibilitó construir de manera sistemática las etapas de entendimiento, recolección, limpieza y ordenamiento de los datos normativos en un corpus que es compatible con el proceso de entrenamiento.  de fine-tuning, adecuando el modelo base al ámbito particular de la normativa universitaria y optimizando su rendimiento en trabajos de respuesta a preguntas, los hallazgos demuestran que el asistente virtual puede producir respuestas coherentes y en tiempo real, mejorando de manera significativa la consulta de normativas por la comunidad universitaria. En síntesis, el prototipo es una contribución innovadora en el ámbito académico porque no solo actualiza la forma de acceder a información normativa, sino que además sienta las bases para la creación de futuros sistemas de soporte institucional fundamentados en inteligencia artificial.



**Palabras clave:** asistente virtual, modelo de lenguaje de gran escala (LLM), fine-tuning, normativa académica, CRISP-DM, procesamiento de lenguaje natural (PLN)

**ABSTRACT**

Access to the regulations of the Faculty of Applied Sciences (FICA) at the Technical University of the North is often an inefficient process, as students and professors must manually review extensive documents, making it difficult to obtain specific information quickly and reliably. To address this issue, this work proposes the development of a virtual assistant based on a large language model (LLM), retrained using fine-tuning techniques, to provide precise and contextualized answers to queries related to institutional regulations. The methodology adopted was based on the CRISP-DM model, which allowed for the systematic structuring of the phases of understanding, collection, cleaning, and organization of the regulatory data into a corpus compatible with the training process. Likewise, a Python development environment was configured, integrating natural language processing libraries such as Hugging Face Transformers and PyTorch, along with optimization techniques to ensure system efficiency. Subsequently, the fine-tuning process was executed, adapting the base model to the specific domain of university regulations and improving its performance in question-answering tasks. The results show that the virtual assistant can generate coherent, reliable, and real-time responses, significantly optimizing the consultation of regulations by the university community. In conclusion, the prototype constitutes an innovative contribution in the academic field, as it not only modernizes access to regulatory information but also sets a precedent for the development of future institutional support systems based on artificial intelligence.

**Keywords:** virtual assistant, large language model (LLM), fine-tuning, academic regulations, CRISP-DM, natural language processing (NLP)

# INTRODUCCIÓN

## TEMA

Desarrollo de un Asistente virtual académico basado en un modelo de lenguaje preentrenado (LLM) para estudiantes de la FICA.

## Problema

### Antecedentes

Históricamente, la administración de las consultas estudiantiles en la Facultad de Ingeniería en Ciencias Aplicadas (FICA) se ha realizado a través de procedimientos manuales, que han sido gestionados completamente por el departamento de secretaría. La necesidad urgente de digitalizar y mejorar los procesos de administración quedó demostrada con la pandemia global reciente. Este contexto ha evidenciado que depender de métodos presenciales es no solo ineficaz, sino también un obstáculo para la calidad y continuidad del servicio, lo cual resalta la necesidad urgente de implementar soluciones tecnológicas en el ámbito educativo.

### Situación Actual

Actualmente, los estudiantes de la FICA enfrentan dificultades significativas para acceder a información académica actualizada y relevante de manera oportuna. El proceso centralizado y manual de consultas genera una sobrecarga de trabajo para el personal de secretaría, lo que resulta en demoras y posibles inconsistencias en la información proporcionada. Esta ineficiencia obliga a los estudiantes a invertir tiempo y esfuerzo en desplazamientos y esperas, afectando negativamente su experiencia educativa y la agilidad de los procesos administrativos de la facultad.

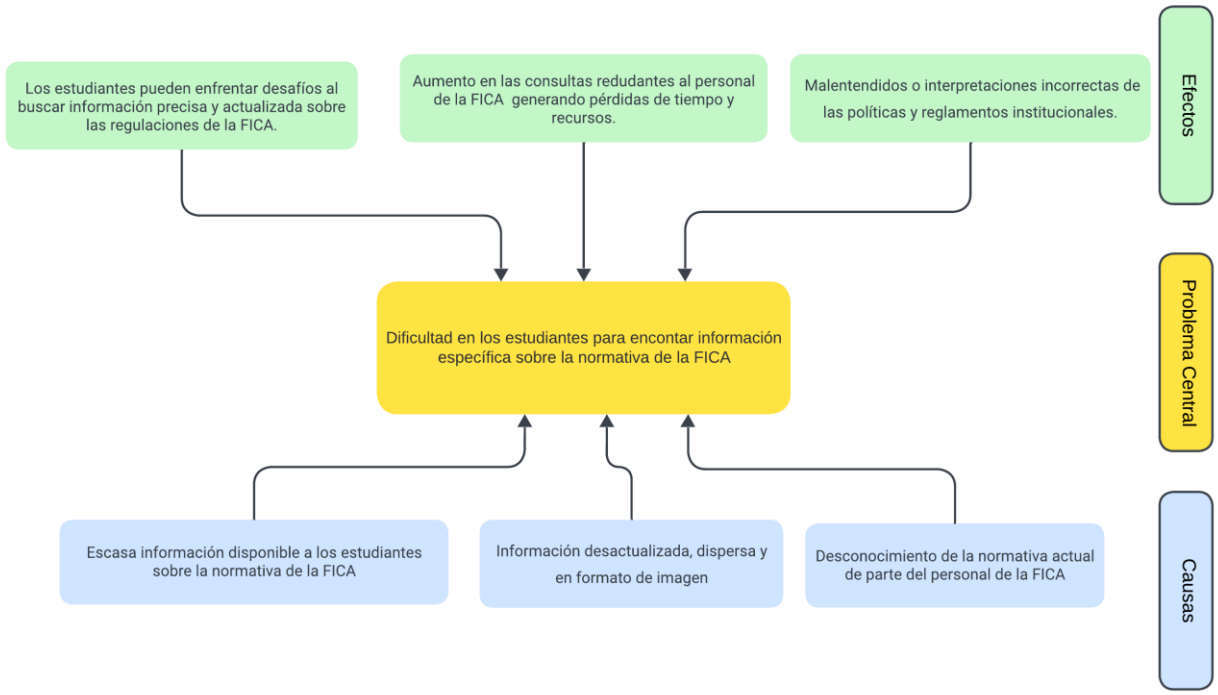
### Planteamiento del Problema

Los alumnos de la Facultad de Ingeniería en Ciencias Aplicadas (FICA) afrontan desafíos significativos en su experiencia educativa, ya que les resulta difícil acceder a información relevante y actualizada. El departamento de secretaría enfrenta una carga significativa debido a que todas las consultas de los alumnos son gestionadas manualmente. La pandemia ha resaltado el valor del entorno virtual como un sistema de gestión fundamental, no únicamente para eludir el contacto físico requerido, sino para optimizar la calidad administrativa de la institución [1]. Es esencial, por ende, el desarrollo y la implementación de un asistente virtual fundamentado en un modelo de lenguaje preentrenado, también llamado modelo de lenguaje grande (LLM). Al ofrecer información instantánea a los alumnos sin la necesidad de trasladarse, se abrirían nuevas

oportunidades y se optimizaría la eficiencia de los sistemas educativos, lo que supondría un ahorro de esfuerzo y tiempo [1]. Este sistema estaría concebido con el fin de cumplir específicamente las necesidades de los alumnos de la FICA. La Figura 1 muestra un árbol de problemas que refleja las causas y consecuencias de esta circunstancia.

**Fig. 1.**

*alcance del proyecto*




Fuente: autoría propia

## Objetivos

### Objetivo General

Desarrollar un asistente virtual basado en un modelo de lenguaje preentrenado (LLM) para consultas académicas respecto a la normativa estudiantil de la FICA

### Objetivos Específicos

- Redactar un marco teórico sobre los LLM y su importancia en el ámbito académico.
- Preprocesar datos sobre la normativa estudiantil de la FICA y reentrenar un modelo LLM usando Python y TensorFlow-Keras.

- Validar el modelo LLM usando métricas cuantitativas de rendimiento.

**Alcance**

El presente trabajo tiene como objetivo desarrollar un asistente virtual utilizando Python con el IDE Visual Studio Code y un modelo preentrenado de lenguaje grande (LLM). El modelo LLM será reentrenado usando librerías como PyTorch, TensorFlow y Keras para poder responder preguntas relacionadas con la normativa estudiantil de la FICA, algunas de ellas son: Reglamento de Estudiantes, Reglamento Interno de la Facultad de Ciencias Aplicadas – FICA, Reglamento General UTN.

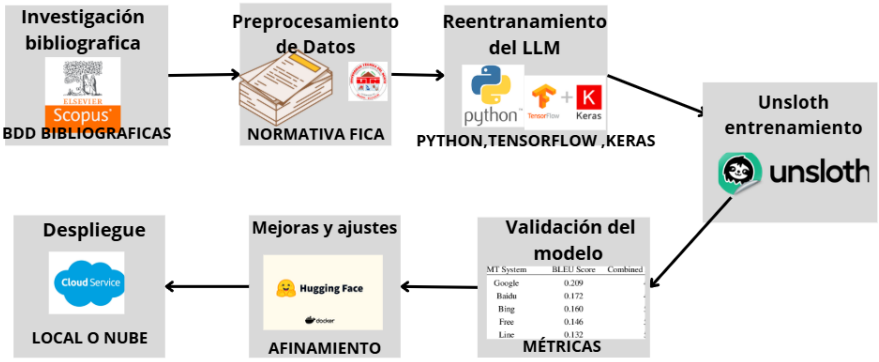
Se llevará a cabo una investigación exhaustiva sobre los diferentes tipos y características de los LLM disponibles con el fin de seleccionar aquel que sea más eficiente para las necesidades del proyecto (benchmarking).

Utilizando la información de la normativa de la FICA, la cual está contenida en documentos en formato PDF (Incluso en formato Imagen) y Word, se preparará y construirá un conjunto de datos (dataset) necesario para reentrenar el modelo LLM. El modelo LLM podrá ser reentrenado usando recursos computacionales o plataformas en la nube como Google colab, Azure, Aws, Kaggle, GCP entre otros.

Una vez completado el proceso de reentrenamiento, se procederá a realizar pruebas exhaustivas del asistente virtual utilizando métricas como BLEU (Bilingual Evaluation Understudy), BLEU es una métrica ampliamente utilizada para evaluar la precisión de las traducciones automáticas, y se puede aplicar para comparar la similitud entre las respuestas generadas por el modelo y las respuestas de referencia [2]. Para asegurar que pueda manejar una variedad de consultas y proporcionar respuestas precisas y coherentes en contextos académicos. A lo largo de este proceso, se harán mejoras y ajustes al modelo LLM según sea necesario. El asistente virtual se podrá instalar en un servidor local o en un servicio de nube. En la Figura 2 se puede observar el alcance del proyecto.

**Fig. 2.**

*Alcance del Proyecto*




*Fuente: Autoría Propia*

**Metodología**

Para el cumplimiento del objetivo 1 se realizará una recopilación y análisis de Información en el primer objetivo, se llevará a cabo una investigación documental para elaborar un marco teórico sobre la importancia de los asistentes virtuales en el ámbito académico utilizando las bases de datos disponibles de la universidad, para el cumplimiento de los objetivos 2 y 3 se basará en la metodología de análisis de datos CRISP-DM:

2. **Preparación de Datos y Reentrenamiento del Modelo LLM** Para el segundo objetivo, se empleará documentación en varias formas (PDF, Word, etc.), que contendrá información pertinente como reglamentos, políticas y procedimientos. Esta información se usará para formar un conjunto de datos que será empleado en el reentrenamiento del modelo LLM. Para perfeccionar nuestro modelo según las necesidades del asistente virtual, este procedimiento se realizará empleando técnicas de Transfer Learning y Fine-Tuning con la biblioteca Transformers de Higgins Face, así como herramientas como TensorFlow, Keras, Python y PyTorch.
3. **Realización de Pruebas y Evaluación de Calidad:** El tercer objetivo consiste en llevar a cabo análisis minuciosos del modelo, empleando métricas como BLEU o evaluaciones humanas para calibrar la calidad de las respuestas producidas por el modelo preentrenado. Con el fin de adquirir una visión cualitativa acerca de la coherencia y la importancia de las respuestas. Se asegurará que el modelo pueda manejar una diversidad de consultas y dar

respuestas coherentes y precisas en contextos académicos.

**Justificación**

Este proyecto aborda un problema actual y relevante en el ámbito académico al integrar tecnologías avanzadas de procesamiento del lenguaje natural (NLP) y modelos de lenguaje LLM preentrenados para mejorar la accesibilidad y comprensión de la normativa por parte de los estudiantes. Además, el proyecto estará alineada con los siguientes objetivos nacionales e internacionales.

**Justificación Educativa**

La implementación de un asistente virtual en el contexto educativo impacta directamente en la calidad de la educación. Al proporcionar una herramienta que facilita el acceso a información precisa y actualizada sobre la normativa estudiantil, se promueve una educación más informada y eficiente. Esto contribuye a mejorar la experiencia educativa de los estudiantes y fomenta el cumplimiento del ODS 4 [3].

**Justificación Tecnológica**

El desarrollo de este asistente virtual supone una innovación tecnológica que favorece el desarrollo de la infraestructura académica y de la industria. El empleo de modelos de lenguaje preentrenados y tecnologías de NLP es un avance en el uso de inteligencia artificial para solucionar problemas concretos en el ámbito educativo. Esto está de acuerdo con el ODS 9, que fomenta la innovación y el desarrollo de infraestructuras tecnológicas fuertes [3].

# CAPÍTULO I

# 1. MARCO TEÓRICO

## 1.1 Variables del tema

Para el desarrollo aprobado de “Desarrollo de un Asistente virtual académico basado en un modelo de lenguaje preentrenado (LLM) para estudiantes de la FICA.”, las variables serán:

El concepto y definición de modelos de lenguaje preentrenados (LLM), incluyendo sus características y funcionamiento, así como su historia y evolución.

Las aplicaciones de los LLM en el sector educativo, subrayando las ventajas que su implementación tiene en contextos académicos y ejemplos concretos de su uso en entidades educativas.

Herramientas y frameworks para el desarrollo de LLM, incluyendo frameworks populares como PyTorch y TensorFlow, así como herramientas y bibliotecas suplementarias como Django, Hugging Face y Unsloth.

La evaluación y las métricas de LLM, que incluyen tanto el análisis de los métodos de evaluación de modelos lingüísticos como el uso de métricas frecuentes como METEOR, BLEU y ROUGE, así como la relevancia del examen constante.

Por último, se examina el efecto que tienen los LLM en la educación, investigando su influencia sobre el aprendizaje y la enseñanza, así como su uso como asistentes virtuales educativos.

Estos factores nos facilitarán la comprensión del contexto y las particularidades de los LLM, además de su implementación y aplicación como ayudantes virtuales en el entorno académico de la FICA, y contribuirán a determinar su eficacia y valor dentro del ámbito educativo.

## 1.2 Inteligencia Artificial y Procesamiento del Lenguaje Natural (NLP)

### 1.2.1 Definición de Inteligencia Artificial

El objetivo de la inteligencia artificial es crear sistemas capaces de realizar tareas que normalmente requerirían inteligencia humana. El aprendizaje, el razonamiento, la percepción, la resolución de problemas, la comprensión del lenguaje natural y la toma de decisiones son algunas de estas tareas. La IA se divide en dos categorías principales: la IA débil o estrecha, diseñada para realizar una tarea específica (por ejemplo, asistentes virtuales como Siri), y la IA fuerte o general, que busca realizar cualquier tarea cognitiva que un ser humano pueda hacer [4].

El término "Inteligencia Artificial" ha cambiado y ha sido interpretado de múltiples formas a lo

largo del tiempo, desde su primera aparición hasta la actualidad. Este término se expande cada vez más debido al crecimiento de esta área. Las normas anteriores para definir la inteligencia artificial se están quedando obsoletas a medida que avanza la tecnología. La inteligencia artificial evoluciona constantemente en beneficio de una amplia gama de industrias. Por ejemplo, ya no se considera que las máquinas que calculan funciones básicas o reconocen texto a través del reconocimiento óptico de caracteres incorporen inteligencia artificial, ya que esta función ahora se da por sentada como una función informática inherente, la inteligencia artificial evoluciona continuamente para beneficiar a muchas industrias diferentes [4].

### 1.2.2 Evolución del Procesamiento del Lenguaje Natural

Tiene sus raíces en la década de 1950, cuando los científicos de la computación comenzaron a explorar formas para enseñar a las máquinas a comprender y producir lenguaje humano. A lo largo de los años, el NLP ha evolucionado a través de diferentes enfoques, incluyendo el NLP basado en reglas, estadístico y híbrido. Actualmente, el enfoque basado en machine Learning es el más popular [5].

1. **Inicios 1950-1960:**

Alan Turing sugirió el Test de Turing en 1950 con el propósito de analizar la inteligencia de una máquina. Los sistemas de traducción automática que se basaban en reglas básicas y diccionarios aparecieron durante los primeros años. Uno de los primeros chatbots que imitaba a un psicoterapeuta, ELIZA, fue creado en 1966 [6].

2. **Sistemas basados en reglas 1970-1980:**

El desarrollo de gramáticas formales y parsers permitió avances significativos en el procesamiento del lenguaje natural. Los sistemas expertos utilizaban reglas predefinidas para entender y generar lenguaje, aunque enfrentaban limitaciones significativas, como la dificultad para manejar ambigüedades y contextos [6].

3. **Giro estadístico 1990-2000**

La implementación de modelos N-gram y HMM (Modelos Ocultos de Markov) en tareas como el reconocimiento de voz marcó el inicio del aprendizaje automático, que empezó a hacer uso de amplios corpus textuales para identificar patrones. Técnicas avanzadas como SVM (Support Vector Machines) y árboles de decisión también emergieron, lo que aceleró aún más el progreso en este campo [7].

4. **Era del Deep Learning 2010**

El avance de representaciones vectoriales de palabras, utilizando métodos como Word embeddings (GloVe, Word2Vec), ha sido esencial. La atención y los Transformers, que se introdujeron en 2017, establecieron las bases para modelos avanzados como GPT y BERT. Por otro lado, las redes neuronales recurrentes (RNNs, LSTMs) fueron empleadas para modelar secuencias. Los modelos preentrenados de gran tamaño, como BERT, T5 y GPT, han tenido un progreso importante en actividades como preguntas y respuestas, traducción y resumen [6].

5. **Modelos multimodales 2020**
6. La inteligencia artificial ha visto incrementadas sus capacidades gracias a la integración de texto con imágenes y voz, como lo demuestran los modelos DALL-E y Whisper. El aprendizaje de cero disparos (zero-shot learning) y el de pocos disparos (few-shot learning) son métodos que posibilitan la adaptación a nuevas tareas con escasos o sin ejemplos. No obstante, todavía subsisten retos significativos, por ejemplo: la privacidad, los sesgos, el consumo de energía y la capacidad de interpretación [6].

### 1.2.3 Principales aplicaciones de NLP

Las aplicaciones de NLP son varias a continuación enumeramos las más comunes y por lo general en lo que destacan y han demostrado tener gran capacidad:

- **Análisis de texto:** Un tipo de NLP es el análisis de texto, que convierte el texto en datos para su examen. Esto se utiliza en organizaciones de la banca, atención a la salud, manufactura y gobierno para ofrecer mejores experiencias a los clientes, reducir el fraude y mejorar la convivencia social [8].
- **Chatbots y asistentes digitales:** El PLN se utiliza en sistemas GPS operados por voz, asistentes digitales y softwares de conversión de voz a texto para mejorar la interacción entre los usuarios y las computadoras. Estos chatbots pueden responder preguntas y entablar conversaciones de gran alcance, lo que aumenta la satisfacción de los clientes y ahorra recursos valiosos para las empresas [7].
- **Automatización de procesos empresariales:** El PLN ayuda a agilizar y automatizar operaciones de negocio, aumentar la productividad de los empleados y simplificar procesos empresariales críticos. Esto se logra mediante la detección de correo basura y organización de documentos, clasificación de currículos y búsqueda en internet por voz [8].
- **Traducción automática:** Los servicios de traducción en línea emplean NLP para traducir

textos al instante y con mayor efectividad. Esto elimina barreras lingüísticas y facilita la comunicación global [6].

- **Búsqueda avanzada de información:** El PLN se utiliza en la búsqueda de información no estructurada en diferentes sectores, como la búsqueda de patentes, la clasificación de documentos y la extracción de información relevante. Esto permite encontrar documentos más pertinentes para una consulta y ahorrar tiempo en la exploración de información [8].
- **Generación de texto:** La generación de texto, también conocida como generación de lenguaje natural (NLG), produce texto similar al escrito por humanos. Estos modelos pueden ajustarse para producir textos de distintos géneros y formatos, como tuits, blogs e incluso código informático [8].

Existen otras categorías de aplicaciones más centradas en resumir o analizar datos, enumerar o evaluar. Además, esta rama de la informática incluye métodos para conversar con otras personas y para traducir sonido a texto (o viceversa). Un ejemplo de esto último es el uso de chatbots en las conversaciones [9].

### 1.2.4 Desafíos en el procesamiento del lenguaje natural

El NLP resuelve una serie de desafíos técnicos y éticos que requieren soluciones innovadoras y multidisciplinarias. Superar estos desafíos es crucial para el desarrollo de sistemas más precisos, justos y eficientes, que puedan integrarse de manera efectiva en una variedad de aplicaciones y mejorar nuestra interacción con la tecnología [10].

- **Ambigüedad lingüística:** El lenguaje humano es inherentemente ambiguo y puede tener múltiples interpretaciones, lo que dificulta la comprensión precisa del texto [10].
- **Variaciones del lenguaje**: El PLN debe adaptarse a la diversidad de idiomas y dialectos existentes, superar barreras lingüísticas y culturales para garantizar una comprensión precisa en todos los contextos [10].
- **Tratamiento de datos no estructurados**: El PLN se enfrenta a problemas al tratar con datos no estructurados, como textos no etiquetados o con formatos variables [10].
- **Comprensión contextual:** Las máquinas deben ser capaces de comprender el contexto en el que se utiliza el lenguaje para evitar malentendidos [10].
- **Privacidad y ética:** La utilización del PLN implica manejar datos a gran escala, lo cual genera desafíos relacionados con la privacidad y la ética. Para asegurar que este tipo de tecnología se use de manera segura y responsable, es esencial definir políticas y

regulaciones apropiadas [10].

## 1.3 Modelos de Lenguaje Preentrenados (LLM)

### 1.3.1 Definición y características de los LLM

Los LLM son modelos de Inteligencia Artificial desarrollados dentro del campo del NLP. Estos modelos pueden entender y generar texto. Esto significa que codifica una distribución de probabilidad sobre cadenas de palabras, la cual se puede denotar como P (w1...wL). Esta distribución busca aproximar la distribución realizada por un amplio "corpus" de texto en el idioma especifico. En general, gran parte de los LLM están basados en la arquitectura Transformer [11], la Tabla 1 sintetiza las características operativas que definen a esta tecnología.

**Tabla 1.**

*Características de los modelos LLM*

| Característica Distintiva | Descripción e Implicación Técnica |
|---|---|
| **Entendimiento y Generación (NLG)** | Capacidad de generar y procesar texto con una coherencia y un estilo parecidos a los de los humanos. Esto permite que se puedan usar aplicaciones complejas, como la síntesis de información, la redacción creativa y la traducción automática. |
| **Aprendizaje Automático Masivo** | El modelo es entrenado con datos de corpus textuales amplios, lo que le permite aprender patrones gramaticales, sintácticos y semánticos complejos a partir de miles de millones de parámetros. |
| **Naturaleza Auto supervisada** | Aprenden de manera autónoma a predecir el siguiente token en una secuencia sin necesidad de etiquetado manual exhaustivo, lo que facilita su adaptación a diversos contextos lingüísticos. |

| | |
|---|---|
| **Escalabilidad de Datos** | Tienen una arquitectura sólida que les permite procesar y organizar grandes volúmenes de datos no estructurados, asimilando la "estructura" del lenguaje a gran escala. |
| **Versatilidad (Multiusos)** | No se encuentran restringidos a una única función; según la instrucción (prompt) que reciban, un mismo modelo tiene la capacidad de actuar como asistente de código, traductor, generador de resúmenes o chatbot conversacional. |
| **Falibilidad y Sesgos** | No obstante su sofisticación, no son bases de conocimiento infalibles. Pueden replicar prejuicios sociales que se encuentran en sus datos de entrenamiento y son susceptibles a "alucinaciones" (producir información falsa). |

Fuente: [12]

### 1.3.2 Principales modelos de lenguaje preentrenados

Algunos de los diversos modelos desarrollados han tenido un impacto especialmente significativo por su habilidad para gestionar tareas complejas con una coherencia y precisión sin igual. Este artículo se enfoca en cuatro modelos de este tipo que han tenido un impacto duradero en el área de la NLP: T5, BERT, GPT-3 y RoBERTa. Examinaremos sus aportaciones a la investigación, sus rasgos únicos y sus aplicaciones prácticas que han modificado los criterios de la inteligencia artificial.

**GPT-3 (Generative Pre-trained Transformer 3)**

Desarrollado por OpenAI, GPT-3 es uno de los modelos de lenguaje más avanzados y potentes hasta la fecha. Con 175 mil millones de parámetros, puede generar texto de alta calidad, responder preguntas, traducir lenguajes y realizar tareas de procesamiento de lenguaje natural (NLP) complejas con alta precisión. Ha demostrado un rendimiento aún más impresionante en tareas de NLP y ha mostrado la capacidad de aprender de pocas muestras, lo que significa que puede adaptarse a nuevas tareas con solo unos pocos ejemplos de entrenamiento. GPT-3 ha sido utilizado en una amplia gama de aplicaciones prácticas, desde la generación de texto hasta el razonamiento y la comprensión del lenguaje natural [13].

**BERT (Bidirectional Encoder Representations from Transformers)**

Desarrollado por Google AI, BERT es un modelo bidireccional que permite comprender el contexto completo de una palabra en una oración. Ha mejorado significativamente el rendimiento en tareas de (NLP) como la generación de texto, el reconocimiento de entidades nombradas y la respuesta a preguntas. Su arquitectura ha sido la base para muchos otros modelos y ha establecido un nuevo estándar en la comprensión del lenguaje natural [14].

**RoBERTa (Robustly optimized BERT approach)**

RoBERTa, desarrollado por Facebook AI, es una versión mejorada de BERT, que se ha entrenado con una mayor cantidad de datos y ajustes más precisos. Su formación más optimizada y sólida le permite superar a BERT en varias tareas de procesamiento de lenguaje natural (NLP). Se ha empleado extensamente en proyectos de investigación y comerciales, lo que ha fortalecido todavía más el impacto de los modelos fundamentados en transformadores [15].

**T5 (Text-To-Text Transfer Transformer)**

T5, creado por Google Research, enfoca las tareas de procesamiento del lenguaje natural (NLP) como desafíos de conversión de texto a texto, lo que brinda una amplia flexibilidad. Desde la traducción y el resumen hasta la respuesta a preguntas y la producción de texto, ha demostrado un desempeño excepcional en una variedad extensa de tareas de procesamiento del lenguaje natural. Su perspectiva integrada ha tenido un impacto en el diseño de otros modelos y en cómo se llevan a cabo las tareas de procesamiento del lenguaje natural (NLP) [16].

**Llama (Large Language Model Meta AI)**

La familia de modelos Llama, desarrollada por Meta AI, ha sido un hito en la democratización de la inteligencia artificial al proporcionar modelos con pesos abiertos que tienen un rendimiento excepcional. Una de sus versiones, Llama 3.2, sobresale por su eficacia y habilidad de razonamiento en modelos con pocos parámetros (como el modelo de tres billones de parámetros). Esta arquitectura posibilita llevar a cabo procesos de ajuste fino (fine-tuning) y ejecución en hardware de consumo con eficacia, conservando un nivel de respuesta semejante al de modelos más grandes[17].

A manera de síntesis, la Tabla 2 presenta una comparativa técnica de los modelos analizados anteriormente. En este resumen se destacan los desarrolladores, las innovaciones arquitectónicas y las aplicaciones principales de cada modelo.

**Tabla 2.**

*Modelos de Lenguaje de Alto Impacto.*

| Modelo | Desarrollador | Característica Distintiva | Aplicación Clave |
|---|---|---|---|
| **GPT-3** | OpenAI | Arquitectura generativa de 175 mil millones de parámetros. | Generación de texto de alta calidad y aprendizaje *few-shot*. |
| **BERT** | Google AI | Procesamiento bidireccional para contexto completo de palabras. | Estándar en la comprensión profunda del lenguaje natural. |
| **RoBERTa** | Facebook AI | Versión optimizada de BERT entrenada con mayor volumen de datos. | Aplicaciones comerciales de NLP con alta robustez. |
| **T5** | Google Research | Enfoque unificado donde toda tarea es de texto a texto. | Flexibilidad multitaria para traducción, resumen y respuestas. |
| **Llama (3.2)** | Meta AI | Modelo de pesos abiertos optimizado para alta eficiencia. | Implementación en hardware de consumo y dominios especializados. |

Fuente: autoría propia

### 1.3.3 Arquitectura de los LLM

El procesamiento del lenguaje natural (PLN) se ha visto transformado por la tecnología de las redes neuronales profundas, en particular los transformadores, que constituyen la base de la arquitectura de los grandes modelos lingüísticos (LLM) [18].

Los componentes claves de la arquitectura de los LLM:

**Transformadores**:

- Los LLMs modernos, como GPT, BERT y otros, se basan en la arquitectura de transformadores. Introducidos en el paper "Attention is All You Need" por Vaswani et al., los transformadores permiten manejar secuencias de datos de longitud variable de manera

más eficiente que las arquitecturas RNN (Redes Neuronales Recurrentes) y LSTM (Long Short-Term Memory) [19].

**Mecanismo de Atención**:

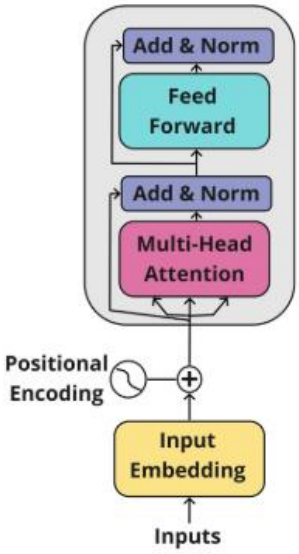
- El mecanismo de atención es crucial para los transformadores. Permite que el modelo enfoque diferentes partes de la secuencia de entrada de manera flexible, asignando pesos a diferentes palabras basándose en su relevancia en el contexto actual. Esto es particularmente útil para capturar relaciones de largo alcance en el texto [18].

**Codificadores y Decodificadores**:

- Los modelos transformadores generalmente consisten en pilas de codificadores y/o decodificadores. BERT, por ejemplo, utiliza solo la pila de codificadores, mientras que GPT utiliza una pila de decodificadores. Los codificadores procesan la secuencia de entrada para generar una representación interna, y los decodificadores utilizan esta representación para producir la salida, a continuación, en la Figura 3, se muestra una ilustración de un codificador.

**Fig. 3.**

*Bloque de Codificador Transformer*

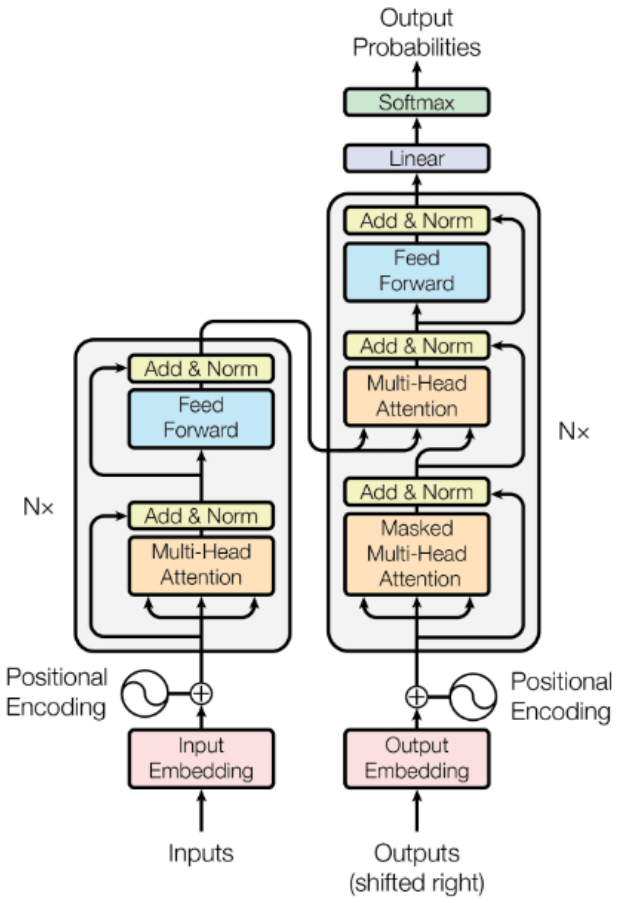


Fuente: [20]

Como podemos ver en la Figura 4 el codificador, situado a la izquierda de la imagen, convierte la secuencia de entrada en datos contextuales. Organiza en datos de contexto. Esta parte no está pensada para predecir texto por sí sola. La siguiente palabra potencial del texto la predice el descodificador situado en el lado derecho utilizando los datos del codificador y la salida producida en el paso anterior. El decodificador de la derecha predice la siguiente palabra de la secuencia utilizando los datos del codificador y el resultado obtenido en el paso anterior. El decodificador de la derecha predice la siguiente palabra potencial de la secuencia utilizando los datos del codificador y el resultado obtenido en el paso anterior. Producida en el paso anterior para predecir la palabra potencial siguiente de la secuencia [20].

**Fig. 4.**

*Modelo Transformer Codificador-Decodificador*




Fuente: [20]

Para ajustar las proporciones del vocabulario, la salida del último bloque del decodificador se proyecta a través de una capa lineal. Se ejecuta mediante una función softmax, que convierte los logits1 en probabilidades de palabra y ajusta las dimensiones a la cantidad del vocabulario inicial. Esto permite al modelo anticipar la palabra siguiente en la secuencia utilizando la información codificada y el contexto [20]. Los transformadores pueden procesar secuencias más largas de manera más eficiente y capturar mejor las relaciones entre las palabras, lo que los hace más efectivos para tareas de procesamiento de lenguaje natural.

La arquitectura de un modelo de lenguaje grande (LLM) opera así: en primer lugar, el modelo recibe el texto inicial y lo segmenta en unidades menores a las que se les conoce como tokens. Después, transforma estos tokens en vectores de números (embeddings) que simbolizan lo que significan. Incorpora información acerca de la ubicación de cada token para conservar el orden del texto. A continuación, emplea mecanismos de atención para examinar la manera en que los tokens se interrelacionan. Este proceso se lleva a cabo en distintas capas, cada una de las cuales mejora la interpretación del texto. Por último, el modelo vuelve a proyectar estos vectores mejorados hacia palabras y estima las probabilidades de las siguientes palabras para producir una salida coherente y pertinente en términos de contexto.

### 1.3.4 Modelo de lenguaje preentrenado Llama

Los LLM son la vanguardia de la IA. Entre ellos, la familia Llama de Meta destaca por su gran importancia y accesibilidad. Estos son modelos de lenguaje de código abierto basados en la arquitectura Transformer, entrenados para comprender y generar texto. La reciente versión Llama 3.2, por ejemplo, ofrece un equilibrio óptimo entre alto rendimiento y eficiencia, con variantes como la de 3 mil millones de parámetros (Llama-3.2-3B-Instruct) afinadas para seguir instrucciones y conversar. Su naturaleza abierta y su soporte multilingüe los hacen herramientas esenciales para la investigación y el desarrollo de aplicaciones innovadoras en Procesamiento del Lenguaje Natural [21]. A continuación, en la Tabla 3 podemos ver la información más importante de uno de los modelos de la familia llama.

***Tabla 3.***

*Caracteristicas de Llama*

| Característica Técnica | Llama 3.2 Estándar | Llama 3.2 Cuantizado |
|---|---|---|

| | | |
|---|---|---|
| **Datos de Entrenamiento** | Nueva mezcla de datos públicos en línea (9T tokens). | Nueva mezcla de datos públicos en línea (9T tokens). |
| **Parámetros Totales** | 1B (1.23B) y 3B (3.21B) | 1B (1.23B) y 3B (3.21B) |
| **Modalidad de Entrada** | Texto Multilingüe | Texto Multilingüe |
| **Modalidad de Salida** | Texto Multilingüe y Código | Texto Multilingüe y Código |
| **Ventana de Contexto** | 128k Tokens | 8k Tokens |
| **Atención (Arquitectura)** | GQA (Grouped Query Attention) | GQA (Grouped Query Attention) |
| **Embeddings Compartidos** | Sí | Sí |
| **Corte de Conocimiento** | Diciembre 2023 | Diciembre 2023 |

Fuente: [21]

## 1.4 Importancia de los LLM en el Ámbito Académico

### 1.4.1 Aplicaciones de los LLM en la educación

La creación de estrategias tecnológicas es crucial, ya que el panorama educativo evoluciona a un ritmo cada vez más rápido. Para cumplir los objetivos de desarrollo en el campo de la educación, son cruciales las estrategias tecnológicas y la investigación sobre la capacidad de los LLM., los cuales tienen diversas aplicaciones específicas que pueden mejorar la eficiencia y la calidad del aprendizaje. En el caso particular de un asistente virtual académico diseñado para responder preguntas sobre la normativa estudiantil, las aplicaciones más relevantes incluyen lo descrito en la Tabla 4 según [22] :

***Tabla 4.***

*Aplicaciones y Beneficios de un LLM como Asistente Virtual*

| Aplicación Específica | Descripción y Beneficio Institucional |
|---|---|
| **Apoyo en la comprensión normativa** | El modelo simplifica el lenguaje jurídico complejo a explicaciones comprensibles, detallando las políticas de evaluación, las reglas de conducta y los requisitos de registro para favorecer el entendimiento del estudiante. |
| **Acceso rápido a información crítica** | Facilita la exploración de datos específicos (fechas límite, requisitos para becas, apelaciones) utilizando lenguaje natural, lo cual disminuye de manera significativa el tiempo de búsqueda en comparación con la inspección manual de documentos PDF. |
| **Orientación en procesos administrativos** | Facilita la asistencia paso a paso en trámites burocráticos complicados (como matrículas y permisos especiales), lo que reduce la cantidad de consultas presenciales en la oficina de administración. |

| | |
|---|---|
| **Ayuda a la toma de decisiones** | Ofrece información exacta sobre los derechos de los estudiantes y los requisitos para graduarse, lo que le permite al estudiante organizar su carrera académica con datos confiables y recientes. |
| **Canal de comunicación institucional** | Actúa como un medio dinámico para la divulgación de actualizaciones normativas y recordatorios sobre plazos críticos, garantizando que los estudiantes estén al tanto de las regulaciones vigentes |
| **Asistencia en la comprensión de la normativa** | El asistente actúa como un "traductor" de términos legales, proporcionando explicaciones detalladas y accesibles sobre reglas académicas, requisitos de inscripción, normas de conducta y políticas de evaluación, facilitando el entendimiento del reglamento oficial. |
| **Acceso rápido a información relevante** | Posibilita que se realicen consultas directas sobre requisitos de becas, fechas límite o apelaciones. Esto facilita el acceso a información esencial que, de manera convencional, requeriría una exhaustiva búsqueda manual en documentos largos. |
| **Asesoría sobre procedimientos administrativos** | Orienta al alumno, por medio de una guía paso a paso, para realizar trámites complejos (como la inscripción en materias, permisos y procesos de titulación). Su puesta en marcha minimiza los errores en la documentación y aligera la carga operativa del personal administrativo. |
| **Apoyo en la toma de decisiones** | Al propiciar que todos tengan acceso a información precisa sobre derechos y obligaciones, los alumnos se empoderan para tomar decisiones fundamentadas acerca de su camino académico, tales como qué materias elegir o cómo planear su graduación. |
| **Mejora en la comunicación institucional** | Actúa como un canal activo, accesible todo el día y todos los días de la semana, para difundir recordatorios sobre plazos críticos y actualizaciones regulatorias. De este modo, se garantiza que los estudiantes estén al tanto y en sintonía con las normativas en vigor. |

Fuente: [22].

Estas aplicaciones particulares posibilitan que el asistente virtual basado en un LLM se convierta en un recurso útil para los alumnos de la FICA. Este asistente optimiza su entendimiento de las regulaciones académicas y favorece su cumplimiento, lo que ayuda a tener una experiencia educativa más organizada y efectiva.

### 1.4.2 Beneficios de utilizar LLM en entornos académicos

El uso de Modelos de Lenguaje Preentrenados (LLM) en entornos académicos presenta varios beneficios específicos, particularmente en el contexto de un asistente virtual diseñado para responder preguntas sobre la normativa estudiantil beneficios que también se encontraron en trabajos de grado como [23] y [24]:

***Tabla 5.***

*Beneficios De Asistentes Virtuales en la Gestión Académica*

| Beneficio Identificado | Impacto en el Entorno Académico |
|---|---|
| **Disponibilidad 24/7 y Acceso Inmediato** | Elimina la dependencia de los horarios de oficina. Los estudiantes obtienen respuestas precisas en tiempo real (fines de semana o madrugadas), resolviendo dudas urgentes sin esperas. |
| **Optimización de Recursos Administrativos** | El personal administrativo tiene la posibilidad de concentrarse en casos difíciles y tareas que tienen un valor estratégico más alto al automatizar las respuestas a preguntas frecuentes (consultas repetitivas), lo cual mejora la eficacia operacional [24]. |
| **Estandarización y Consistencia** | El modelo asegura que todos los alumnos obtengan la misma respuesta coherente y uniforme, eliminando así cualquier contradicción o ambigüedad, lo cual es distinto a la interacción humana, que puede variar de acuerdo al funcionario. |
| **Mejora de la Experiencia Estudiantil (UX)** | Proporciona un entorno de soporte continuo y fácil de usar. Esto incrementa la confianza y seguridad del estudiante, quien se siente respaldado al contar con herramientas modernas para gestionar su vida académica [25]. |
| **Escalabilidad y Actualización Dinámica** | La estructura del modelo hace posible que los cambios en la normativa (nuevas reglas o fechas) se reflejen rápidamente, garantizando que la información proporcionada esté siempre actualizada sin necesidad de reentrenar a gran escala. |

| | |
|---|---|
| **Fomento de la Cultura de Cumplimiento** | El acceso democratizado a normas claras y comprensibles disminuye la fricción cognitiva para comprender las reglas. Esto promueve el cumplimiento voluntario de la normativa y reduce la tasa de infracciones debido a la falta de conocimiento. |

Fuente: autoría propia

Estos beneficios mostrados en la Tabla 5 demuestran cómo los LLM pueden ser una herramienta para mejorar la administración y el apoyo académico dentro de las instituciones educativas. Al facilitar una mejor comunicación y comprensión de la normativa entre los estudiantes de la FICA, los LLM pueden contribuir significativamente a un entorno académico más eficiente y comprensible.

### 1.4.3 Impacto de los LLM en la enseñanza y el aprendizaje

El aprendizaje y uso de las herramientas de inteligencia artificial, surgida durante el siglo XXI, se marca como una tendencia a nivel mundial ha cobrado relevancia en los últimos años, causando gran impacto social en todas las áreas de desarrollo, debido en gran parte al aislamiento durante la contingencia sanitaria por COVID-19 [26]. En la práctica docente, los LLM pueden ayudar a los profesores de la FICA a identificar las áreas donde los estudiantes necesitan más apoyo, permitiendo una intervención oportuna y eficaz. Por ejemplo, si un grupo de estudiantes tiene dificultades con un tema específico de matemáticas, el asistente virtual puede proporcionar recursos adicionales y ejercicios prácticos para mejorar su comprensión. Además, los LLM pueden fomentar el autoaprendizaje, ya que los estudiantes pueden interactuar con asistentes virtuales para resolver dudas y explorar temas de interés a su propio ritmo [27].

## 1.5 Frameworks y Metodología para el Desarrollo del Asistente Virtual

### 1.5.1 Descripción de Frameworks populares

**TensorFlow**

- **Descripción**: Frameworks de código abierto desarrollado por Google, ampliamente utilizado en la investigación y producción de modelos de aprendizaje automático, incluyendo LLM [28].

- **Características**: Posee un amplio conjunto de bibliotecas y herramientas, admite diversas plataformas (CPU, TPU, GPU) y proporciona una alta escalabilidad y flexibilidad.

**PyTorch**

- **Descripción**: PyTorch es un Frameworks de código abierto desarrollado por Facebook. Es muy popular en la comunidad de investigación debido a su facilidad de uso y su enfoque en la programación dinámica [29].
- **Características**: Proporciona un entorno intuitivo y una fuerte integración con Python, facilitando la experimentación y el desarrollo rápido de modelos.

**Hugging Face**

- **Descripción**: Hugging Face es un ecosistema abierto que reúne la librería Transformers (más de 300 arquitecturas con soporte "day-0" para los modelos más actuales), el Hub, para almacenar y versionar conjuntos de datos y pesos, y herramientas como Accelerate y PEFT, las cuales sirven para realizar un entrenamiento distribuido y un ajuste fino eficaz. Su meta es simplificar la investigación y la producción en IA estandarizando el acceso, la adaptación y el despliegue de modelos de audio, lenguaje y visión [30].
- **Características**: Con una API sencilla de utilizar, que facilita la carga y el uso de modelos preentrenados para múltiples funciones de PLN, es extremadamente accesible.

**Django**

- **Descripción**: Django es un framework de desarrollo web de código abierto, escrito en Python, que facilita la creación rápida de aplicaciones web robustas y seguras. Su arquitectura basada en el patrón Model-View-Template (MVT) permite estructurar eficientemente el código, lo cual es especialmente útil al construir sistemas complejos como asistentes virtuales o chatbots inteligente [30].
- **Características**: Django ofrece instrumentos fundamentales para el desarrollo del backend de un chatbot, por ejemplo la integración con APIs, la administración de rutas (URLs), la autenticación de los usuarios y el almacenamiento y obtención de datos a través de su ORM.

- **La creación de endpoints**: Para procesar las preguntas que hacen los usuarios y remitir las respuestas producidas por el modelo de lenguaje.

### 1.5.2 Ventajas y desventajas de cada Frameworks

Para determinar la base tecnológica del proyecto, se realizó un análisis comparativo de los frameworks de aprendizaje profundo más utilizados en la industria y la academia. La selección se basó en criterios de escalabilidad, facilidad de uso y especialización en Procesamiento del Lenguaje Natural (NLP). La Tabla 6 contrapone las fortalezas y debilidades de TensorFlow, PyTorch y el ecosistema de Hugging Face.

***Tabla 6.***

*Ventajas y Limitaciones de cada Framework*

| Framework / Herramienta | Ventajas Principales | Desventajas y Limitaciones |
|---|---|---|
| **TensorFlow** | • Alta escalabilidad y ecosistema robusto de despliegue (TF Serving).<br>• Soporte nativo para hardware especializado (TPU).<br>• Herramientas visuales como TensorBoard. | • Debido a su complejidad arquitectónica, tiene una curva de aprendizaje pronunciada.<br>• Menos capacidad de adaptación para la rápida experimentación en comparación con métodos dinámicos. |

| | | |
|---|---|---|
| **PyTorch** | • Diseño intuitivo ideal para investigación y prototipado ágil.<br>• Amplia adopción en la comunidad académica y facilidad para depurar.<br>• Integración activa de grafos computacionales | • Históricamente, el ecosistema de despliegue en producción (Mobile/Web) ha sido menos maduro que TensorFlow, aunque está mejorando rápidamente. |
| **Hugging Face** (Transformers) | • Especialización absoluta en modelos de lenguaje (LLMs) preentrenados.<br>• API de alto nivel que abstrae la complejidad de PyTorch/TensorFlow.<br>• Acceso a miles de modelos y dataset listos para usar. | • Dependencia de las bibliotecas base (es necesario instalar PyTorch o TF por debajo).<br>• Si se necesita cambiar capas profundas fuera de la arquitectura Transformer estándar, la flexibilidad disminuye. |

Fuente: autoría propia

Después de analizar estas opciones, se eligió una arquitectura híbrida fundamentada en Hugging Face y PyTorch. Esta combinación posibilita beneficiarse de la facilidad y flexibilidad de depuración que ofrece PyTorch, gracias a la gran biblioteca de modelos preentrenados de Hugging Face. Esta elección fue táctica para la puesta en práctica de métodos contemporáneos de ajuste fino, como Unsloth, que se desarrollan sobre el ecosistema de PyTorch por su predominancia en la investigación actual de LLMs.

### 1.5.3 Herramientas y bibliotecas adicionales para el desarrollo de LLM

#### Transformers Library (Hugging Face)

- **Descripción**: Biblioteca especializada en modelos de lenguaje y procesamiento del lenguaje natural (NLP), que brinda acceso a un extenso abanico de modelos ya entrenados.
- **Uso**: Permite cargar, ajustar y utilizar modelos preentrenados en múltiples tareas de procesamiento del lenguaje natural.

**TensorBoard (TensorFlow)**

- **Descripción** Herramienta de visualización para TensorFlow que posibilita observar y rastrear histogramas de pesos, gráficos de modelos y métricas de entrenamiento [31].
- **Uso**: Perfecto para supervisar el avance del entrenamiento y depurar modelos.

**Unsloth**

- **Descripción:** Unsloth es una biblioteca de parameter-efficient fine-tuning (PEFT) que combina adaptadores LoRA/QLoRA, cuantización a 4 bits y kernels CUDA optimizados (Flash-Attention 2) para acelerar de 2 a 5 veces el entrenamiento de modelos de lenguaje y reducir hasta un 80 % la demanda de VRAM. Gracias a estas optimizaciones, permite ajustar modelos de gran tamaño (≥ 8 B parámetros) en GPUs comerciales de 5-8 GB, convirtiéndola en una solución accesible y reproducible para entornos académicos y de investigación con recursos limitados [32].
- **Uso:** En este proyecto, se empleará Unsloth como instrumento fundamental para afinar un modelo Llama-3 8B, evidenciando la posibilidad técnica de ajustar LLMs con hardware sencillo. Se cargará específicamente el modelo en 4 bits, se habilitarán los adaptadores LoRA (con r = 64) y se entrenará usando el entrenador incorporado de Unsloth [32].

Estas bibliotecas y herramientas contribuyen a los principales Frameworks, añadiendo características que simplifican el desarrollo, la evaluación, el entrenamiento y la implementación de modelos de lenguaje preentrenados (LLM).

### 1.5.4 Protocolo WebSockets vs. HTTP

Para garantizar la interactividad del asistente virtual, es imperativo seleccionar un protocolo de transporte que minimice la latencia. Si bien HTTP es el estándar para la transferencia de documentos estáticos, su modelo de petición-respuesta resulta ineficiente para aplicaciones que requieren un flujo de datos continuo. Investigaciones recientes en el ámbito de

la tecnología educativa[33], han validado la arquitectura basada en WebSockets sobre HTTPS/TLS como el estándar para sistemas que requieren una sincronización, la Tabla 7 contrasta las características de ambos protocolos, justificando la elección técnica basada tanto en la teoría como en la literatura aplicada consultada.

***Tabla 7.***

*Comparativa técnica entre HTTP Y WebSockets*

| Característica | HTTP (REST) | WebSockets (WSS) | Justificación basada en la Literatura [Cita] |
|---|---|---|---|
| **Persistencia** | Conexión efímera. Se cierra tras cada respuesta. | Conexión persistente y encriptada (TLS). | Estudios sobre monitoreo en tiempo real demuestran que mantener el canal abierto es vital para gestionar flujos continuos (alertas o tokens) sin la sobrecarga de reconexión. |
| **Direccionalidad** | Unidireccional (El cliente solicita algo y el servidor responde) | Bidireccional (full-duplex). | Facilita que el servidor envíe información al cliente (por ejemplo, una palabra del chatbot o una alerta de examen) sin tener que aguardar a que este último haga alguna pregunta. |
| **Caso de Uso Ideal** | Carga de páginas web o peticiones estáticas a bases de datos. | Aplicaciones con baja latencia (monitoreo, videojuegos, chats). | La puesta en marcha de sistemas de alertas académicas instantáneas demuestra que WebSockets es la estructura sólida para contextos en los que el tiempo de respuesta es fundamental. |

Fuente: autoría propia

### 1.5.5 CRISP-DM (Cross-Industry Standard Process for Data Mining)

La metodología CRISP-DM (Cross Industry Standard Process for Data Mining) es un modelo de referencia ampliamente aceptado para la planificación y ejecución de proyectos de minería de datos y ciencia de datos, CRISP-DM se caracteriza por su enfoque iterativo, flexible y orientado al negocio, lo que permite adaptar el proceso de análisis de datos a distintos contextos organizacionales, académicos y tecnológicos. Debido a estas características, según [34] se ha consolidado como una de las metodologías más utilizadas tanto en entornos empresariales como en proyectos académicos y de investigación a continuación en la Tabla 8 vemos las fases.

**Tabla 8.**

*Fases del ciclo de vida de CRISP-DM*

| Fase del Ciclo | Definición Teórica y Objetivos |
|---|---|
| **1. Comprensión del Negocio** | Es la etapa inicial en la que los objetivos de la organización se convierten, desde el punto de vista técnico, en un problema relacionado con minería de datos. Se definen los estándares de éxito y se examinan los recursos existentes. |
| **2. Comprensión de los Datos** | Consiste en recopilar, describir y analizar inicialmente los datos sin procesar. Su propósito es detectar dificultades de calidad (sesgos, valores nulos) y hallar primeros conocimientos (insights) acerca de los datos. |
| **3. Preparación de los Datos** | Conocida como la fase más costosa en términos de tiempo (generalmente el 60-80% del proyecto). Incluye la limpieza, la transformación, la elección de características y la organización del conjunto de datos final para el modelado. |
| **4. Modelado** | Fase en la que se eligen y utilizan las técnicas de modelado (como por ejemplo Árboles de Decisión, Redes Neuronales). Los hiperparámetros se calibran aquí y se crean los modelos matemáticos iniciales. |
| **5. Evaluación** | El modelo se somete a un examen exhaustivo de carácter técnico (métricas de error) y comercial antes de su lanzamiento, con el fin de garantizar que soluciona la cuestión planteada en la primera fase. |

| | |
|---|---|
| **6. Despliegue** | Integración del conocimiento adquirido o del modelo entrenado dentro de los procesos operativos de la organización, que puede ir desde un reporte hasta una aplicación de software interactiva. |

Fuente: [35]

La adopción de CRISP-DM en investigaciones modernas sobre modelos de lenguaje se justifica por la naturaleza experimental del Fine-Tuning. Autores contemporáneos como [35] señalan que el entrenamiento de modelos no es un proceso lineal de entrada y salida, sino un ciclo de refinamiento constante. La flexibilidad de CRISP-DM permite transitar fluidamente entre la Preparación de Datos y el Modelado, facilitando la corrección de formatos y la limpieza de corpus textuales

## 1.6 Evaluación y Métricas de LLM

### 1.6.1 Métodos de evaluación de modelos de lenguaje

Evaluar modelos de lenguaje puede ser un desafío considerable debido a su alta complejidad. La evaluación es fundamental para garantizar la precisión y efectividad de estos modelos en una variedad de aplicaciones. Métodos como la perplejidad, la pérdida de entropía cruzada y métricas como BLEU y ROUGE permiten medir cómo los modelos manejan tareas lingüísticas específicas y su desempeño en contextos prácticos. Además, la evaluación humana juega un papel crucial al juzgar aspectos como la coherencia, fluidez y relevancia del texto generado. A medida que los modelos de lenguaje y aprendizaje automático más avanzados se desarrollan, es crucial que la evaluación evolucione para abarcar criterios amplios que aseguren su solidez, eficacia, equidad, interpretabilidad, impacto y seguridad en contextos específicos. Las evaluaciones recientes se han enfocado en diversos aspectos clave [36] explicados en la Tabla 9:

***Tabla 9.***

*Dimensiones críticas en la evaluación de Modelos LLM*

| Dimensión de Evaluación | Descripción y Relevancia |
|---|---|

| | |
|---|---|
| **Robustez y Generalización** | Capacidad del modelo para mantener un rendimiento consistente frente a entradas imprevistas, ruidosas o fuera de la distribución de entrenamiento asegurando fiabilidad en escenarios reales. |
| **Equidad y Sesgo** | Reconocimiento y atenuación de prejuicios sociales presentes en los datos de entrenamiento. El objetivo es impedir que el modelo mantenga estereotipos discriminatorios o produzca contenido que ofenda. |
| **Interpretabilidad y Explicabilidad** | Desafío de abrir la "caja negra" de los LLMs. Se refiere al desarrollo de mecanismos que permitan entender cómo y por qué el modelo llega a una conclusión específica, aumentando la confianza del usuario. |
| **Impacto Ambiental y Eficiencia** | Evaluación del coste computacional, consumo energético y huella de carbono. Esta dimensión prioriza el uso de modelos optimizados (como SLMs) frente a arquitecturas masivas insostenibles. |
| **Precisión Específica de Tarea** | Medición del desempeño en dominios especializados (médico, legal, traducción). Evalúa no solo la fluidez lingüística, sino la exactitud técnica y cultural del contenido generado. |
| **Evaluación Centrada en el Humano** | Incluye juicios cualitativos humanos para evaluar dimensiones subjetivas que las métricas automáticas pasan por alto, tales como la seguridad, la coherencia de la narración, la utilidad práctica y la empatía. |

Fuente: [36]

### 1.6.2 Métricas comunes (BLEU, ROUGE,Etc)

La evaluación de modelos de lenguaje es fundamental para medir su desempeño y asegurar que

cumplan con los estándares de calidad requeridos en diversas aplicaciones. Las métricas automatizadas, como BLEU, ROUGE y METEOR, junto con la evaluación humana, son herramientas esenciales para este propósito. Cada una de estas métricas proporciona una perspectiva única sobre la calidad del texto generado, permitiendo a los investigadores y desarrolladores identificar fortalezas y áreas de mejora en sus modelos, la métrica BLEU (Bilingual Evaluation Understudy) es uno de los métodos más utilizados para evaluar la calidad de las traducciones automáticas. Introducida por Kishore Papineni y sus colegas en 2002, BLEU se ha convertido en un estándar en el campo de la traducción automática [37], la precisión basada en n-gramas es el primer componente de BLEU. Una subsecuencia de una secuencia dada se denomina n-grama. Por ejemplo, para una frase dada “El cielo está nublado en comparación al cielo de ayer” [38].

**Fig. 5.**

*Ejemplo de Unigramas*

| Unigramas | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| El | cielo | está | nublado | en | comparación | al | cielo | de | ayer |

Fuente:[38]

la lista de bigramas o 2-gramas sería

**Fig. 6.**

*Segundo ejemplo de Unigrama*

| Bigramas | | | | |
|---|---|---|---|---|
| El cielo | cielo está | está nublado | nublado en | en comparación |
| comparación al | al cielo | cielo de | de ayer | |

Fuente:[38]

La precision1 en el candidato 1 sería (1 +2+1+1+0+0+1)/10=6/10=0.6,
mientras que en el segundo candidato sería (1+2+1+1+1+1+1)/10 = 8/10 = 0.8.
La precision2 en el candidato 1 sería (1 +0+0+0+0+0)/9=1/9=0.11,
mientras que en el segundo candidato sería (1+0+0+0+0+1+0)/9=2/9=0.22.
Para calcular el BLEU se toma la media geométrica de todas las precisiones tal

como vemos en la Ecuación.

**Fig. 7.**

*Formula de BLEU*

$$Precision_{BLEU} = (\prod_{i=1}^{4} precision_i)^{\frac{1}{4}}$$

Fuente:[38]

**Concepto Básico**

BLEU mide la similitud entre una oración generada automáticamente (por ejemplo, por un modelo de lenguaje) y una o más oraciones de referencia escritas por humanos. La idea es que cuanto más se parezca el texto generado al de referencia, mejor será la calidad de la generación, ROUGE (Recall-Oriented Understudy for Gisting Evaluation) es una métrica de evaluación utilizada en el NLP, particularmente para la evaluación de sistemas de resumen automático. ROUGE, que fue creada en 2004 por Chin-Yew Lin, tiene como objetivo evaluar la calidad de los resúmenes producidos por máquinas mediante la comparación con uno o más resúmenes de referencia (elaborados por humanos). El sistema que se centra en la habilidad de capturar el contenido informativo fundamental del texto original es evaluado por esta métrica.

La evaluación hecha por humanos es un método de los más exactos y fiables para evaluar la calidad de las traducciones automáticas que produce un LLM. A pesar de que las métricas automáticas como METEOR y BLEU son ventajosas por su rapidez y consistencia, no logran captar la riqueza ni todos los matices del lenguaje humano. Por esta razón, la evaluación humana continúa siendo una práctica fundamental y habitual en el desarrollo e investigación de sistemas de traducción automática [39].

### 1.6.3 Bert Score para evaluación de semántica

Una de las métricas más usadas para la evaluación y coherencia de las respuestas del asistente virtual por lo tanto se ha considerado la métrica BERTScore como uno de los métodos automáticos para evaluar la calidad de las respuestas generadas por el modelo de lenguaje. Esta métrica ha sido propuesta como una alternativa más robusta a las métricas tradicionales como

BLEU o ROUGE, ya que permite medir la similitud semántica entre dos textos, más allá de la coincidencia superficial de palabras o frases.

El BERTScore emplea representaciones vectoriales de las palabras que producen los modelos preentrenados tipo Transformer, como el BERT (Bidirectional Encoder Representations from Transformers). BERTScore utiliza los embeddings contextuales de cada palabra, lo que permite captar el significado en su contexto y mejorar la evaluación para que se asemeje más a la percepción humana de similitud explicado en la Tabla 10. Esto contrasta con las métricas n-grama como BLEU, que dependen de coincidencias exactas entre tokens [40].

***Tabla 10.***

*Flujo de funcionamiento algorítmico de la métrica BERTScore.*

| **Fase del Proceso** | **Descripción Técnica** | **Objetivo** |
|---|---|---|
| **1. Tokenización Contextual** | Las oraciones (la generada por el chatbot y la referencia humana) se descomponen en unidades mínimas o *tokens* utilizando el vocabulario del modelo preentrenado. | Organizar el texto conservando la estructura gramatical para su análisis. |
| **2. Generación de Embeddings** | Una red Transformer (BERT/RoBERTa) procesa cada token para adquirir su representación en forma de vector. Estos, a diferencia de los vectores estáticos, varían dependiendo del contexto de la oración. | Registrar el sentido semántico profundo de cada palabra en función de sus palabras vecinas. |
| **3. Cálculo de Similitud** | Para crear una matriz de alineación par a par, se determina la similitud de coseno entre cada token de la respuesta generada y cada token de referencia. | Calcular la distancia matemática entre los significados de las palabras |

| | | |
|---|---|---|
| **4. Greedy Matching y Agregación** | Para determinar los resultados finales de F1, Precision y Recall, el algoritmo escoge las parejas de tokens que más se asemejan (emparejamiento voraz). | Conseguir una métrica única que castigue la incoherencia, pero no sancione el empleo de sinónimos correctos. |

Fuente: [41]

**Funcionamiento de BERTScore**

El proceso de cálculo de BERTScore se resume en los siguientes pasos:

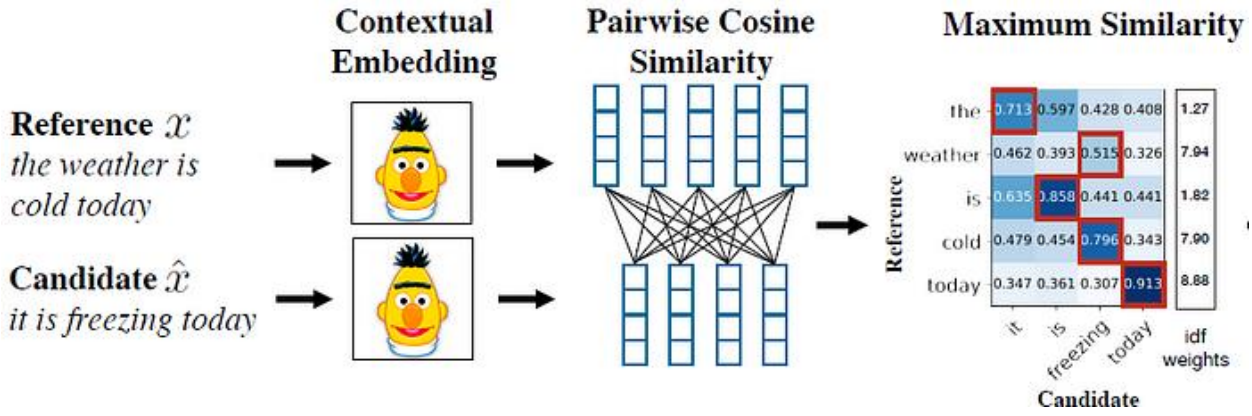
- **Tokenización** de las oraciones de referencia y de las respuestas generadas.
- **Extracción de embeddings contextuales** para cada token, utilizando un modelo BERT preentrenado.
- **Cálculo de la similitud de coseno** entre los tokens de la oración generada y los tokens de la oración de referencia.

Determinación de tres métricas fundamentales:

- **Precisión (P):** Qué tan bien los tokens generados semánticamente se corresponden con los de la referencia.
- **Recall (R):** Cuánto de la información que contiene la referencia es capturada por la respuesta producida.
- **F1 Score:** Promedio armónico entre la precisión y el recall, que muestra la proporción entre los dos.

**Fig. 8.**

*Computacion de la métrica Bert score*




Fuente:[42]

La Figura 8 ilustra el proceso de cálculo de BERT Score, una métrica avanzada diseñada para evaluar la similitud semántica entre una oración de referencia (X) y una oración candidata (X^). Este método inicia con la incrustación contextual (Contextual Embedding) de ambas oraciones mediante un modelo pre-entrenado como BERT, el cual tokeniza el texto en "piezas de palabra" y genera representaciones vectoriales que capturan el significado de cada token dentro de su contexto oracional. Posteriormente, se calcula la similitud coseno por pares (Pairwise Cosine Similarity) entre cada vector de la referencia y cada vector de la candidata, generando una matriz de similitudes. Para determinar la máxima similitud (Maximum Similarity), se aplica una coincidencia codiciosa (*greedy matching*), donde cada token de la referencia se empareja con el token más similar de la candidata, y viceversa para la precisión. Opcionalmente, se pueden incorporar pesos IDF (frecuencia inversa de documento) para ponderar la importancia de las palabras raras. Finalmente, BERT Score combina estas medidas de recuperación (Recall) y precisión (Precision) para obtener un F1-Score, ofreciendo una evaluación robusta de la similitud semántica que supera las limitaciones de las métricas de coincidencia superficial [42].

### 1.6.4 Perplexity

La perplejidad constituye una métrica intrínseca fundamental en la evaluación de modelos de lenguaje probabilísticos, derivada de la teoría de la información para cuantificar el grado de incertidumbre o sorpresa que experimenta el modelo al predecir la siguiente unidad léxica en una secuencia. Matemáticamente definida como la exponenciación de la entropía

cruzada media, esta métrica evalúa la capacidad del sistema para modelar la distribución de probabilidad del corpus de evaluación; en términos prácticos, un valor reducido de perplejidad indica que el modelo asigna altas probabilidades a las palabras objetivo, reflejando una mayor fluidez sintáctica y adaptación al dominio lingüístico, mientras que valores elevados denotan una incapacidad para anticipar la estructura del texto[43]. No obstante, la literatura especializada advierte que, si bien una baja perplejidad es condición necesaria para la coherencia gramatical, no es suficiente para garantizar la veracidad factual, por lo que su interpretación debe realizarse conforme a los criterios expuestos en la Tabla 11

***Tabla 11.***

*Valores De Perplejidad*

| **Nivel de Perplejidad** | **Interpretación Técnica** | **Comportamiento Esperado del Modelo** |
|---|---|---|
| **Baja (Ideal)** | El modelo presenta una elevada seguridad en sus predicciones y escasa incertidumbre. | Produce texto que sea gramaticalmente correcto, fluido y que mantenga coherencia con el contexto. Con precisión elevada, predice la palabra que sigue. |
| **Alta** | El modelo enfrenta una gran incertidumbre ("sorpresa") frente a los datos. | Produce secuencias sin coherencia, que contienen errores de gramática o saltos lógicos. Significa "adivinar" entre un número excesivo de opciones posibles. |
| **Divergente (Infinita)** | El modelo no ha conseguido converger a lo largo de la capacitación. | Salida de texto completamente aleatoria o repetitiva sin sentido semántico. |

Fuente: autoría propia

### 1.6.5 Evaluación Automatizada y Humana

Es esencial evaluar los modelos de lenguaje para asegurar su desempeño y satisfacer los estándares de calidad en una variedad de aplicaciones. Para este fin, son herramientas esenciales las métricas automatizadas como ROUGE y BLEU, además de la evaluación humana.

Cada una ofrece una perspectiva única sobre la calidad del texto generado, permitiendo a investigadores y desarrolladores identificar tanto fortalezas como áreas de mejora en sus modelos[38].

BLEU, por ejemplo, se destaca como una métrica estándar en la evaluación de traducciones automáticas, utilizando la precisión de n-gramas para medir la similitud entre el texto generado y las referencias humanas. Su introducción en 2002 por Kishore Papineni y colaboradores ha sido fundamental para el avance en la calidad de la traducción automática [39].

Además, BERT Score es esencial para evaluar la calidad de las respuestas generadas por el modelo. En contraste con métricas convencionales que se fundamentan en la coincidencia precisa de palabras clave o n-gramas, como ROUGE o BLEU, BERT Score utiliza las incrustaciones contextuales de modelos de lenguaje extensos (por ejemplo, BERT) para calcular la similitud semántica en profundidad entre la respuesta del chatbot y una respuesta ideal de referencia, Esto es crucial para un chatbot conversacional, porque hace posible analizar si la respuesta no solo incluye palabras pertinentes, sino también si verdaderamente comprende la intención y el significado de la referencia, incluso cuando emplea una formulación distinta. En consecuencia, BERT Score ofrece una valoración más exacta y correlacionada con el ser humano de la adecuación, coherencia y fluidez de las contestaciones del chatbot, lo que es esencial para el proceso de afinamiento y perfeccionamiento incesante del sistema [42].

**Tabla 12.**

*Comparación de Métricas*

| Métrica | Evalúa semántica | Apta para español | Sensible a sinónimos | Fácil de implementar |
|---|---|---|---|---|
| **BLEU** | No | Sí | No | Sí |
| **ROUGE** | No | Sí | No | Sí |
| **BERT Score** | Sí | Sí | Sí | Sí |
| **Evaluación humana** | Sí | Sí | Sí | No (requiere tiempo) |

Fuente: autoría propia

La combinación de estos métodos explicados en la Tabla 12 brinda una perspectiva completa para perfeccionar incesantemente los modelos de lenguaje y asegurar su utilidad en aplicaciones prácticas y críticas [38], esta combinación de herramientas evaluativas garantiza que los modelos de lenguaje no solo sean exactos desde un punto de vista técnico, sino que además tengan la capacidad de generar resultados que concuerden eficientemente con las expectativas y necesidades del usuario final.

## 1.7 Trabajos relacionados

### Trabajo 1

Según [24], la implementación de un asistente virtual (CHATBOT) para el blog de la Carrera de Software de la Universidad Técnica del Norte utilizando inteligencia artificial, mediante el uso de diagflow, Python y validando con el modelo de DeLone & McLean, el 98% de los participantes expresaron que el chatbot mejoró la eficiencia. A posteriores mejoras o implementaciones se debe tener en cuenta usar un modelo el cual trabaje sin usar Key-words así mejorado la funcionalidad del chatbot y reduciendo aún más el trabajo de la secretaria de la carrera de software.

### Trabajo 2

Propuso [23] , Implementar un chatbot como estrategia de apoyo en el servicio de soporte bibliotecario de la Universidad Técnica del Norte aplicando técnicas de procesamiento de lenguaje natural, utilizando Microsoft Bot Framework, lenguaje C# y el entorno de desarrollo Visual Studio con el patrón MVC (Modelo-Vista-Controlador) usando el modelo de éxito de DeLone y McLean para evaluar el desempeño del chatbot como resultados se obtuvo que la mayoría de los encuestados (53%) estuvo de acuerdo en que el chatbot había mejorado la productividad, mientras que un número similar (44%) estuvo de acuerdo en que había mejorado la accesibilidad y un número similar (36%) dijo que estaba muy descontento con cómo había mejorado la eficiencia. En general, los datos demuestran que los usuarios estaban satisfechos con el chatbot basado en inteligencia artificial. A futuro se puede explorar el uso de un LLM para mejorar la experiencia de usuario y la efectividad del chatbot.

**Trabajo 3**

Reyes y Castillo [44] desarrollaron un asistente virtual para mejorar el acceso a la información académica y administrativa de la FIEC - ESPOL. Se diseñó una arquitectura con Lang Chain por su comunidad y mantenimiento. Se eligieron los modelos Chat GPT y Llama2 por su alta precisión. La información se obtuvo mediante scraping de la web de FIEC. Se desarrolló una interfaz web con Streamlit para el despliegue del servicio. Las encuestas mostraron que el prototipo era 40% más rápido y brindaba una mejor experiencia de usuario en comparación con la web de FIEC. se recomienda investigar a fondo los modelos adecuados, sobre todo en las métricas de evaluación utilizando métricas de recall como BLEU y ROUGE,

**Trabajo 4**

La inteligencia artificial aplicada en la innovación educativa en el proceso de enseñanza y aprendizaje [45], estudio realizado por Ronquillo y Rodríguez mediante una revisión de la literatura, entrevistas a expertos, estudios de caso, experimentos controlados y análisis de datos para evaluar el impacto de la inteligencia artificial en la educación. El artículo identificó nuevas áreas de investigación y aplicaciones de IA en la educación, evaluó su impacto en el rendimiento académico, y analizó implicaciones éticas y pedagógicas. También ofreció recomendaciones para futuras investigaciones y prácticas, destacando la necesidad de colaboración interdisciplinaria y la participación de educadores y estudiantes. En este artículo se puede considerar la exploración de nuevos enfoques de inteligencia artificial, como el aprendizaje automático, el procesamiento del lenguaje natural o la visión por computadora, para ampliar el alcance de la investigación y las posibles aplicaciones en la educación.

**Trabajo 5**

En su tesis [46] se enfrentan al reto de desarrollar asistentes virtuales que puedan llevar a cabo tareas ambiguas en contextos realistas. Crearon y evaluaron un agente basado en un Modelo Grande de Lenguaje (LLM) a través de un estricto procedimiento de ciencia del diseño, en dos rondas, junto con la compañía Destiny Sweden Service Center. El estudio, que empleó un conjunto

de datos de 82 instrucciones ambiguas, evidenció que, aunque el modelo base mostró un desempeño pobre, las mejoras introducidas tuvieron un impacto significativo a nivel estadístico. El incremento del tamaño del modelo (de 7B a 72B parámetros) fue el elemento más decisivo, ya que aumentó significativamente la tasa de éxito y la comprensión de lo que el usuario pretendía. Sin embargo, el rendimiento se vio notablemente mejorado gracias a optimizaciones en el diseño del sistema como la inclusión de restricciones de planificación y la mejora de la claridad de las instrucciones, especialmente con el modelo más pequeño. El estudio determina que una combinación de un modelo de lenguaje grande y un diseño de software detallado es capaz de gestionar la ambigüedad con eficiencia, lo que hace viable este tipo de sistemas y da paso a futuras investigaciones sobre arquitecturas multiagente más complejas y la experiencia del usuario.

**Trabajo 6**

Luna [47] propuso la implementación de un chatbot basado en modelo de lenguaje de inteligencia artificial para responder preguntas frecuentes de estudiantes universitarios usando Node.js y Express para el backend, con una REST API que facilita la transferencia de datos. MongoDB Atlas se utilizó para la base de datos, y la API de WhatsApp Cloud permitió enviar y recibir mensajes mediante un webhook. Además, se integró la API de OpenAI con GPT-3.5 Turbo, presentando un nivel de usabilidad aceptable con un puntaje promedio CUQ (Chatbot Usability Questionnaire) de 73,4. Además, se descubrió que el 93% de los participantes pensaba que las respuestas del chatbot eran beneficiosas. Se puede implementar una interfaz web en la cual el usuario ingrese mediante la web y no únicamente por WhatsApp esto mejoraría la disponibilidad de la aplicación y deber tomado en cuenta para futuros trabajos.

**Trabajo 7**

El siguiente trabajo realizado por [48] la optimización (fine-tuning) de modelos de lenguaje (LLM) para su aplicación como agentes de lenguaje, buscando superar las limitaciones del "few-shot prompting" que resultan en bajo rendimiento, altos costos y latencias. Se propuso FireAct, un enfoque de fine-tuning utilizando trayectorias de agentes generadas por GPT-4 a partir de múltiples tareas de preguntas y respuestas (QA) y métodos de "prompting" (como ReAct, CoT y Reflexion), unificadas en formato ReAct, y con acceso a la API de búsqueda de Google. Se llevaron a cabo

evaluaciones de LMs como CodeLlama, GPT-3.5 y Llama-2. Los resultados mostraron que el fine-tuning incrementa de manera constante el desempeño de los agentes: HotpotQA experimentó un incremento del 77% con Llama2-7B y del 25% con GPT-3.5. Asimismo, mejoró la robustez frente a herramientas ruidosas y disminuyó en un 70% el tiempo de inferencia (de 9.0s a 2.7s) de GPT-3.5, se detectaron beneficios de generalización para nuevas tareas y se descubrió que al combinar métodos y tareas en el fine-tuning, la diversidad de datos optimizó adicionalmente la eficacia y versatilidad del agente. Según el estudio, el fine-tuning es más apropiado para situaciones de uso de explotación a gran escala. Para trabajos futuros, se aconseja utilizar FireAct en una gama más amplia de tipos de tareas y configuraciones de interacción con el ambiente, además de investigar la mejora de agentes más sofisticados que desempeñan varios roles y contextos. Además, se recomienda investigar el ajuste fino a múltiples tareas en gran escala utilizando modelos de última generación y crear benchmarks más variados para los agentes.

**Trabajo 8**

En el siguiente trabajo realizado por [49] se analiza los enfoques y técnicas para el ajuste fino (fine-tuning) de Modelos Lingüísticos Grandes (LLMs), centrándose en su aplicación en el campo de la medicina. El estudio explora un flujo de trabajo general y profundiza en metodologías clave como el ajuste fino supervisado, el aprendizaje por refuerzo con retroalimentación humana (RLHF), métodos de parámetros eficientes (PEFT) como QLoRA, y la generación aumentada por recuperación (RAG). Si bien estas técnicas ofrecen beneficios significativos como la mejora del conocimiento del dominio y la optimización del rendimiento en tareas específicas, la investigación también subraya limitaciones críticas, incluyendo el riesgo de "alucinaciones" (información incorrecta), los sesgos en los datos, y serias preocupaciones sobre la seguridad y privacidad del paciente. Por lo tanto, el estudio concluye que, aunque los LLMs están preparados para transformar la medicina, es imperativo superar estos desafíos para garantizar que las futuras aplicaciones sean robustas, seguras y verdaderamente fiables para guiar a los profesionales de la salud en el mundo real.

**Trabajo 9**

La investigación que llevó a cabo [50] examina la percepción y satisfacción de 391 alumnos con un asistente virtual basado en OpenAI ChatGPT 3.5, el cual se implementó en 21 materias de una universidad online. Se estableció que el "rendimiento del asistente" es el indicador

más relevante de la satisfacción del alumnado, con un análisis de comentarios y una metodología mixta que empleó el cuestionario validado COMUNICA. Se corroboró este hallazgo a través de un Análisis Factorial Confirmatorio (AFC), además de determinarse cuantitativamente. El impacto en el aprendizaje y la mejora de habilidades vienen después. Esta perspectiva fue fortalecida por los hallazgos cualitativos, con un 66% de opiniones favorables que destacaron la ventaja de la herramienta y el deseo de ampliar su empleo. En términos generales, la investigación concluye que estos asistentes mejoran el aprendizaje autónomo y tienen un efecto positivo en la educación superior a distancia. Sin embargo, se detectan desafíos técnicos y se propone como futura línea de investigación realizar un seguimiento longitudinal e incorporar la visión del docente para enriquecer los resultados.

**Trabajo 10**

La inteligencia artificial (IA) se ha transformado en una herramienta dinámica y accesible para el proceso de enseñanza-aprendizaje en la educación superior, según se muestra en el siguiente trabajo de[51], a partir de un análisis bibliográfico y documental. La investigación muestra que la inteligencia artificial brinda soluciones novedosas, como personalizar el aprendizaje y automatizar tareas por medio de asistentes virtuales y chatbots, que transforman los papeles convencionales de maestros y alumnos y optimizan la experiencia en la educación. No obstante, se detectan también retos significativos, como la desigualdad y la brecha de acceso, los temores en torno a la privacidad y ética de los datos, así como la necesidad de formación constante para el profesorado. Para finalizar, aunque la inteligencia artificial brinda una posibilidad de innovación educativa y de generación de un aprendizaje más versátil e interactivo como nunca antes, todavía no se ha alcanzado todo su potencial teórico; por lo tanto, es fundamental establecer procedimientos de regulación para asegurar el cumplimiento de los derechos fundamentales y reducir los peligros vinculados a su aplicación.

# CAPITULO II

# 2. DESARROLLO

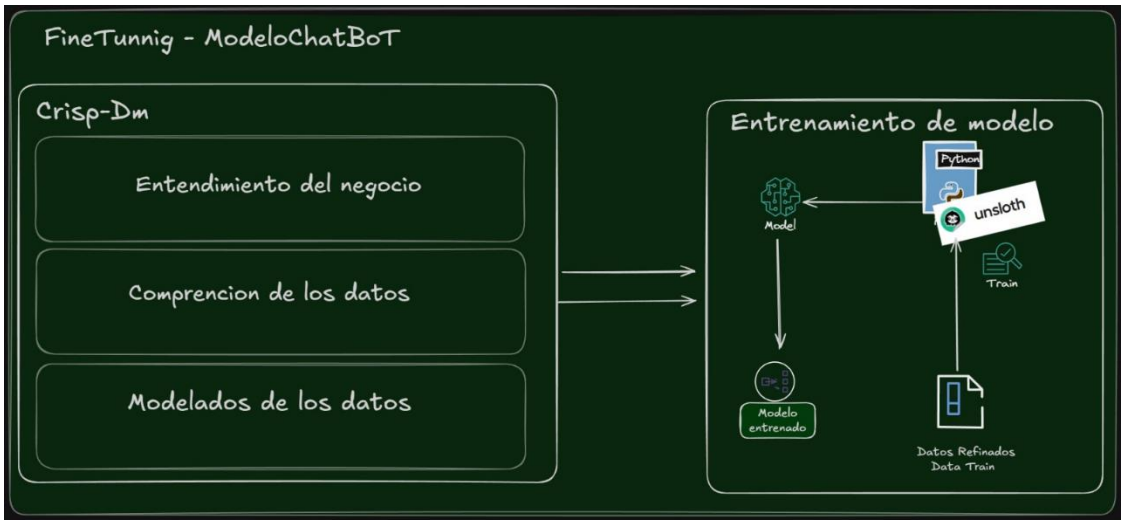
En el presente proyecto se busca presentar el proceso de desarrollo del asistente virtual basado en un modelo de lenguaje preentrenado (LLM) diseñado para responder preguntas relacionadas con la normativa de la Facultad de Ingeniería en ciencias Aplicadas (FICA). Este asistente hace uso de técnicas avanzadas de procesamiento del lenguaje natural (NLP) y se centra en la integración y especialización del modelo mediante el ajuste fino (fine-tuning) con datos específicos extraídos de la normativa institucional, con el objetivo de generar respuestas precisas, coherentes y contextualizadas.

Para alcanzar este propósito, se realizó un proceso de Fine Tuning del modelo base, utilizando un conjunto de pares pregunta-respuesta derivados de documentos oficiales, lo cual permite adaptar el conocimiento del modelo a un dominio específico. A lo largo de este capítulo se describe en detalle la preparación y preprocesamiento de los datos, la configuración del entorno de desarrollo, el proceso de fine-tuning del modelo y las estrategias implementadas para evaluar la calidad, precisión y eficiencia del asistente virtual.

Para asegurar un desarrollo bien organizado, que se pueda repetir y que esté en sintonía con los estándares del sector para proyectos de ciencia de datos, las fases previamente mencionadas se organizaron de acuerdo con un ciclo de vida iterativo. Esta metodología posibilita la vinculación entre las necesidades de la institución y la implementación técnica. El diagrama de flujo general que se diseñó para este proyecto se ilustra en la Figura 9. Este diagrama funciona como una hoja de ruta para las partes siguientes de este capítulo, explicando en detalle el camino desde el entendimiento del problema hasta la ejecución de la solución.

**Fig. 9.**

*Fine Tunning del Modelo*




Fuente: autoría propia

## 2.1 Preparación de Datos

La preparación de datos o dataset, teniendo en cuenta que la normativa y los documentos relacionados con esta se encuentran en formato imagen o formatos no compatibles con el formato requerido se realizó una selección y compresión de los documentos más importantes con la información de la normativa necesaria para poder reentrenar nuestro modelo

### 2.1.1 Identificación de las Fuentes de Información

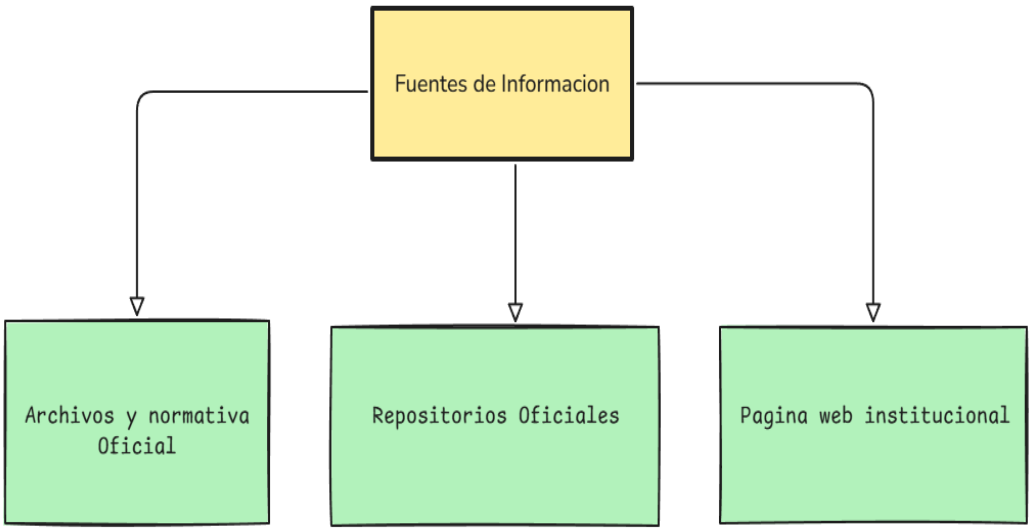
Entre los documentos recopilados se incluyeron reglamentos académicos, manuales operativos, instructivos internos, políticas de vinculación, normativas de prácticas preprofesionales y otros textos complementarios emitidos por la FICA. Estos documentos se encontraban en su mayoría en formato PDF, aunque también se localizaron versiones impresas o escaneadas en formato de imagen, lo cual implicó retos adicionales en cuanto a su digitalización y procesamiento posterior explicado en la Figura 10.

La recopilación de estos documentos se realizó a través de distintas fuentes, entre las cuales se destacan:

- La página web institucional de la FICA.
- Archivos proporcionados directamente por docentes y personal administrativo.
- Repositorios internos a los que se tuvo acceso con fines académicos.

**Fig. 10.**

*Principales Fuentes de Informacion*




Fuente: Autoría propia

La correcta identificación de estas fuentes fue esencial para asegurar que el modelo fuera entrenado con información relevante, actualizada y alineada al contexto real de aplicación, garantizando así la utilidad del asistente virtual en escenarios reales de consulta por parte de estudiantes y docentes.

### 2.1.2 Selección de Documentos Relevantes

Para garantizar la calidad y pertinencia del corpus de entrenamiento, se aplicó un proceso de curaduría documental basado en la relevancia temática y la vigencia institucional. Este filtro permitió separar la información crítica de aquella redundante u obsoleta. La Tabla 13 detalla los criterios de inclusión y exclusión aplicados para definir el conjunto final de documentos, el criterio principal para la selección fue la frecuencia y relevancia temática de los contenidos en el contexto institucional. Se priorizaron los documentos que tratan aspectos como:

***Tabla 13.***

*Criterios de selección y filtrado del corpus documental*

| Criterio de Selección | Temáticas o Características | Justificación Metodológica |
|---|---|---|

| | | |
|---|---|---|
| **INCLUSIÓN (Prioritarios)** | • Régimen académico (matrículas, evaluaciones, asistencia).<br>• Prácticas preprofesionales y vinculación.<br>• Derechos y obligaciones estudiantiles.<br>•Reglamentos de titulación y gestión administrativa. | Se eligieron documentos que, según los estudiantes, tienen una gran demanda de consulta y que las autoridades académicas han validado como actuales y esenciales para el recorrido universitario. |
| **EXCLUSIÓN (Descartados)** | • Documentos derogados o normativas obsoletas.<br>• Textos repetitivos o con escaso contenido normativo.<br>• Procedimientos administrativos internos (sin impacto estudiantil). | Se eliminaron para optimizar la ventana de contexto del modelo con datos útiles, prevenir ilusiones a partir de información antigua y disminuir el "ruido" durante el entrenamiento. |

Fuente: autoría propia

La selección final comprendió un conjunto equilibrado de textos que cubren tanto aspectos generales como específicos de la normativa de la FICA, asegurando así un corpus de entrenamiento representativo y alineado a los objetivos funcionales del asistente virtual.

### 2.1.3 Conversión a Formatos Procesables

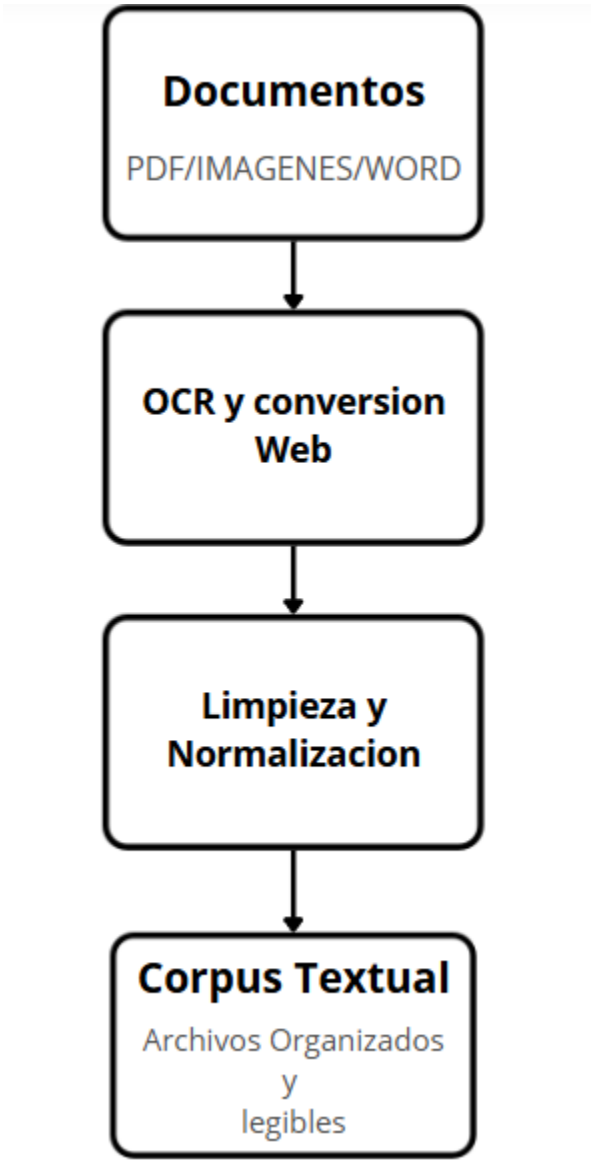
Después de elegir los documentos normativos, se detectó que era necesario cambiar su contenido a formatos que fueran compatibles con el proceso de análisis y entrenamiento del modelo de lenguaje. La mayor parte de los archivos estaba en formato PDF, sin una estructura definida para su lectura automática, o bien eran imágenes escaneadas que necesitaban reconocimiento óptico de caracteres. A causa de estas restricciones, se emplearon instrumentos web de reconocimiento óptico de caracteres (OCR), que posibilitaron la extracción del contenido textual con rapidez y dentro de un margen de error tolerable, sobre todo en documentos impresos con alta calidad.

El texto obtenido fue exportado y posteriormente depurado mediante herramientas de edición manual y scripts de limpieza que eliminaron errores comunes, como caracteres mal reconocidos, saltos de línea innecesarios, y encabezados repetitivos. En los casos donde los documentos ya eran digitales, se utilizaron servicios de conversión de PDF a texto o a formatos editables como Word, lo cual facilitó su manipulación y revisión.

Por último, se normalizaron todos los textos a un formato homogéneo y coherente que no incluía elementos visuales como tablas, imágenes o firmas y que empleaba la codificación UTF-8 y una estructura plana. Esta transformación fue esencial para asegurar que el modelo de lenguaje procesara el contenido normativo durante su fase de entrenamiento, lo que hizo posible trabajar con datos homogéneos, limpios y preparados para ser segmentados semánticamente en unidades funcionales. A continuación, la Figura 11 explica cómo se lleva a cabo la conversión desde los documentos originales hasta el corpus textual disponible para ser entrenado:

**Fig. 11.**

*Flujo de conversión de documentos normativos*




Fuente: Autoría propia

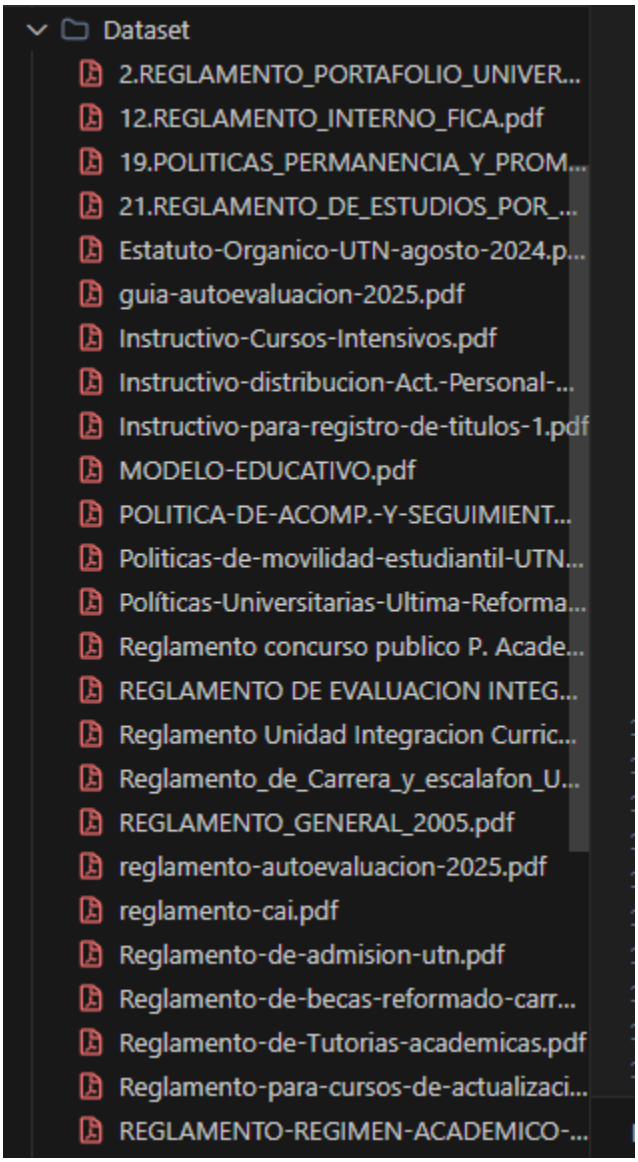
### 2.1.4 Organización del Corpus Normativo

Una vez finalizado el proceso de conversión y limpieza textual, fue necesario organizar el contenido normativo en una estructura que facilitara su uso durante el entrenamiento supervisado del modelo de lenguaje. Para ello, se diseñó un sistema de almacenamiento basado en carpetas temáticas, en las que se agruparon los documentos según su contenido normativo predominante,

tales como “Evaluación Académica”, “Prácticas Preprofesionales”, “Vinculación con la Sociedad” y “Régimen Disciplinario”, entre otras.

**Fig. 12.**

*Dataset Seleccionado*




Fuente: autoría propia

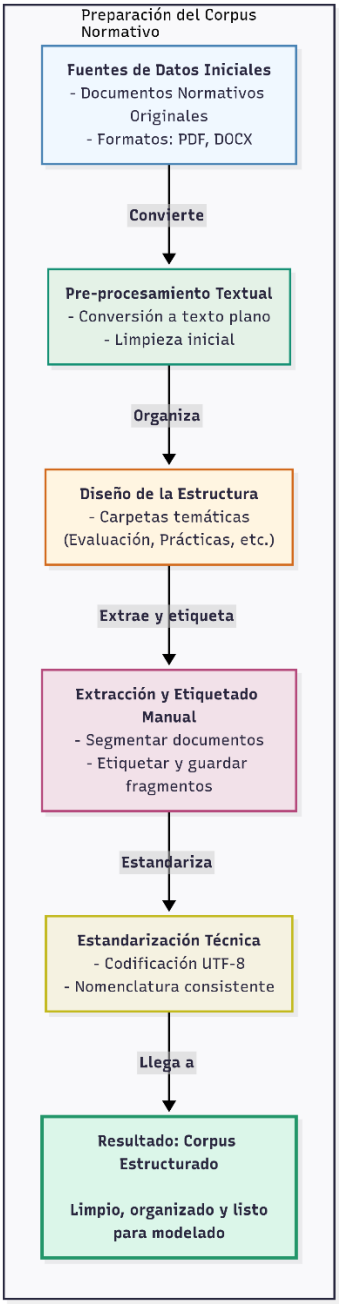
En la Figura 12 se muestra cada archivo dentro de estas carpetas contiene fragmentos textuales coherentes y completos, extraídos de los documentos originales y etiquetados manualmente según su categoría temática. Esta organización permitió mantener un control preciso sobre la distribución de los datos y facilitó su posterior segmentación en pares de entrada-salida

para el entrenamiento del modelo. Además, se empleó una nomenclatura coherente para nombrar los archivos y una codificación uniforme en UTF-8, lo que garantizó su adecuada lectura automática y la posibilidad de rastrearlos durante las pruebas.

La metodología CRISP-DM se utilizó en las etapas posteriores del proyecto, y se fundamentó principalmente en este corpus organizado, lo que posibilitó progresar hacia un procesamiento más estructurado y enfocado en la creación de conocimiento a través del aprendizaje automático.

**Fig. 13.**

*Proceso Preparación Corpus Normativo*



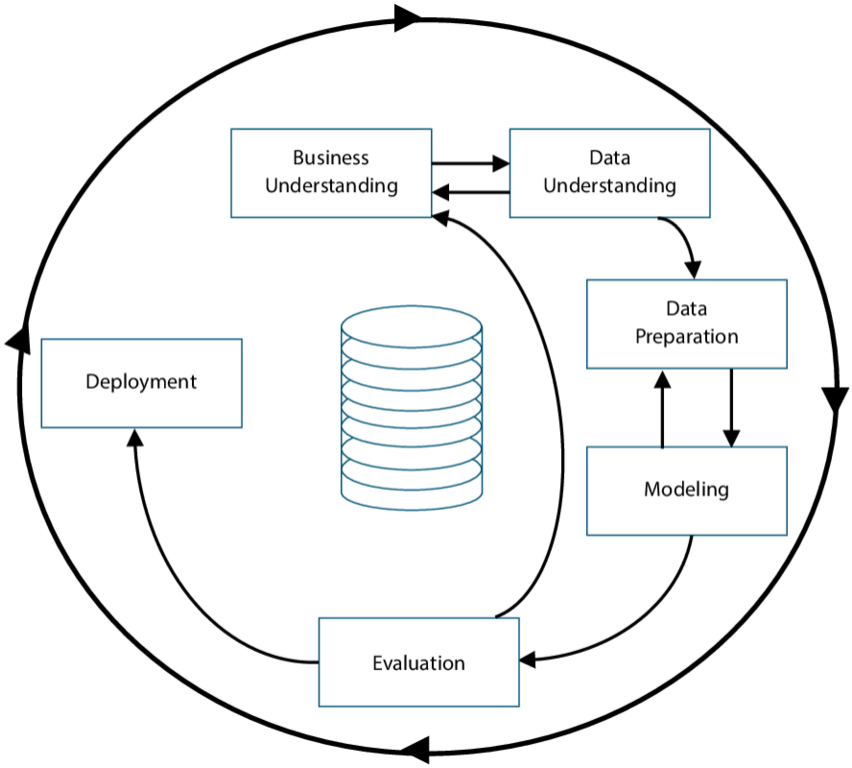

Fuente: Autoría propia

## 2.2 Recopilación y Preprocesamiento de Datos sobre la Normativa de la FICA con la metodología CRISP-DM.

Se utilizó la metodología CRISP-DM (Cross Industry Standard Process for Data Mining) como marco de referencia para dirigir el proceso de preparación de datos que se iba a utilizar para reentrenar el modelo de lenguaje, una vez que se había organizado y estructurado la base documental. Esta técnica, que se utiliza con frecuencia en proyectos de minería de datos y aprendizaje automático, tiene seis etapas: entendimiento del negocio, entendimiento de los datos, preparación de los datos, modelado, evaluación y despliegue explicado en la Figura 14.

**Fig. 14.**

*Metodologia CRISP-DM*




Fuente:[35]

En el contexto de este proyecto, CRISP-DM permitió estructurar de forma sistemática las tareas necesarias para transformar el corpus normativo en un conjunto de datos útil para el aprendizaje supervisado. A diferencia de la etapa anterior, centrada en la obtención y organización física de los archivos, esta fase se enfocó en dar sentido estructurado y semántico a la información, asegurando su coherencia y funcionalidad como insumo para el modelo.

En esta sección se explica el proceso de adecuación y aplicación de las etapas de CRISP-DM al caso concreto del asistente virtual para la normativa de la FICA, empezando por el entendimiento del ámbito normativo y progresando hasta la creación de pares pregunta-respuesta que se emplearán en el ajuste fino del modelo.

### 2.2.1 Compresión del Negocio

El propósito principal de este proyecto es crear un asistente virtual que sea capaz de contestar, de manera exacta y automática, preguntas vinculadas a la normativa actual de la Facultad de Ingeniería en Ciencias Aplicadas (FICA). Este propósito tiene como finalidad satisfacer una necesidad institucional auténtica: optimizar el acceso rápido y fiable a la información normativa para estudiantes, profesores y personal administrativo.

La normativa académica de la FICA es extensa, dispersa en múltiples documentos y su comprensión puede representar una dificultad para los usuarios no familiarizados con el lenguaje técnico-administrativo. Por ello, la implementación de un modelo de lenguaje que procese preguntas en lenguaje natural y entregue respuestas concretas y contextualizadas representa una innovación significativa en el entorno educativo, durante esta fase, se definieron con claridad los siguientes elementos clave del negocio:

- Usuarios objetivo: estudiantes, docentes y personal académico.
- Tipo de información solicitada: reglamentos internos, titulación, vinculación, prácticas preprofesionales y normativa vinculada al régimen académico.
- Funcionalidad esperada del asistente: análisis de preguntas en lenguaje natural y provisión de respuestas exactas, comprensibles y recientes, sin ambigüedades.

Esta comprensión inicial facilitó la definición de los límites del proyecto, la determinación de criterios para la elección y disposición del corpus normativo y el direccionamiento de las etapas posteriores de preparación y modelado del sistema.

### 2.2.2 Comprensión de los Datos

Una vez establecido el objetivo del proyecto y delimitado el dominio normativo que el asistente virtual debía manejar, se procedió a analizar las características del conjunto de documentos recopilados. Esta fase, correspondiente a la “comprensión de los datos” dentro de la metodología CRISP-DM, permitió identificar las particularidades del corpus normativo de la FICA y definir las estrategias adecuadas para su posterior procesamiento.

El corpus se compuso sobre todo de documentos PDF y de versiones escaneadas en formato de imagen. Estos documentos constaban de reglamentaciones académicas, manuales operativos, instrucciones para la vinculación, regulaciones sobre prácticas preprofesionales y otros textos producidos por la facultad. Aunque el contenido era formal y estructurado, se observaron diversas cualidades importantes que tuvieron impacto en las decisiones técnicas:

- Lenguaje técnico-jurídico: Para evitar ambigüedades en las respuestas generadas por el modelo, se necesitaba un tratamiento semántico meticuloso al utilizar términos del ámbito normativo académico.
- Diversidad de contenido: Se encontraron documentos que presentaban un alto nivel de superposición temática (como múltiples textos que regulan temas semejantes, tales como prácticas o titulación), lo cual requería una selección minuciosa para prevenir repeticiones innecesarias.
- Longitud y densidad de información: la dificultad para extraer pares de pregunta-respuesta sin perder contexto se debía a que algunos textos trataban una variedad de temas en secciones largas.

Con el propósito de tratar estas especificidades y reducir los errores comunes que provienen del reconocimiento óptico (desalineación o palabras incompletas), se creó un motor de reglas determinista en Python. En esta fase, se evitó la utilización de LLMs externos para asegurar la exactitud de los datos. La estrategia de extracción se llevó a cabo en dos etapas:

- OCR y lectura híbrida: Se estableció un procedimiento de extracción doble por medio de pdfplumber para la lectura directa de PDFs digitales y pytesseract como apoyo para aplicar OCR en documentos escaneados, garantizando que se pueda recuperar información aun en archivos con baja calidad visual.
- Eliminación de ruido: Se crearon scripts para eliminar patrones repetidos que no brindaban valor semántico, como encabezados institucionales y pies de página, para no "ensuciar" el contexto del modelo.

### 2.2.3 Preparación de los Datos

Con base en la comprensión previa del corpus normativo y sus características lingüísticas y estructurales, se procedió a la fase de preparación de los datos. Esta etapa consistió en transformar el contenido recopilado en un formato adecuado para su uso en el reentrenamiento del

modelo de lenguaje, asegurando la coherencia semántica, la integridad sintáctica y la relevancia normativa de la información, el proceso de preparación de datos abarcó las siguientes actividades clave:

- Segmentación del contenido normativo: Se localizaron y separaron partes textuales que contenían información de manera autónoma, las cuales generalmente eran secciones completas, artículos, incisos o cláusulas. Esta segmentación posibilitó el trabajo con unidades temáticas consistentes, lo que ayudó a crear ejemplos de entrenamiento.
- Estrategia de sinterización de pares de pregunta-respuesta (QA): Se utilizó una estrategia de diversificación para prevenir el sobreajuste, empleando los fragmentos seleccionados. En vez de crear una sola pregunta por artículo, se produjeron automáticamente tres versiones de entrenamiento para cada unidad normativa, con el objetivo de abarcar diversos niveles de interacción cognitiva como se explica en la Tabla 14.

***Tabla 14.***

*Estrategias para el Dataset*

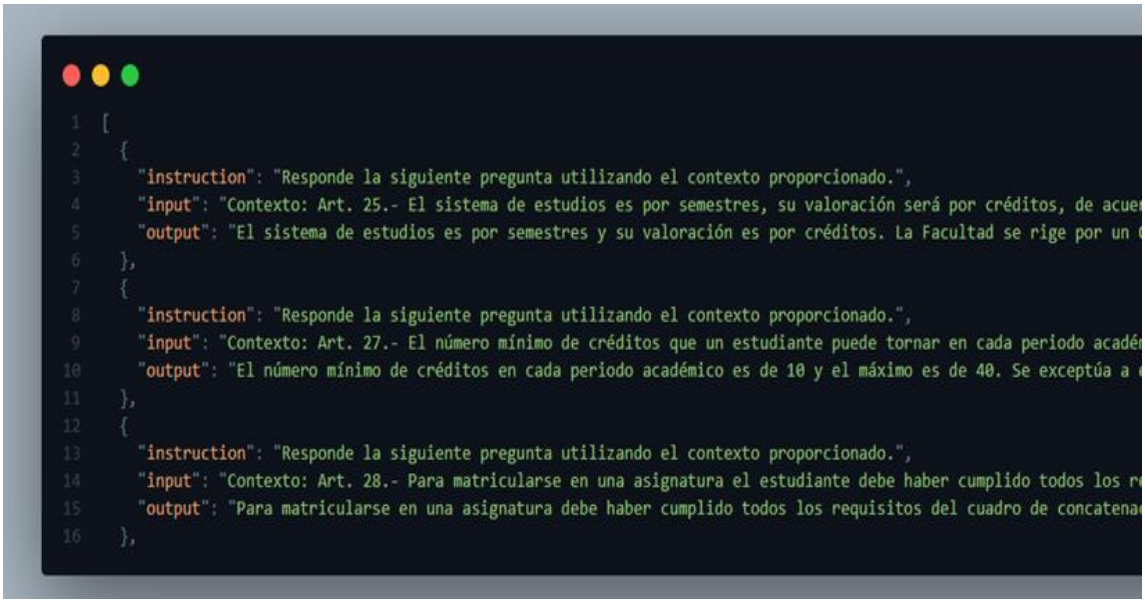
| Estrategia de Variación | Objetivo de Entrenamiento (Fine-Tuning) | Ejemplo de Prompt Generado |
|---|---|---|
| **Variante Directa** | **Memorización de Referencia:** Entrena al modelo para recuperar información exacta basada en identificadores numéricos o códigos específicos de la normativa. | *"¿Qué dice el Artículo 45?"* |
| **Variante Temática** | **Asociación Conceptual:** Fomenta la búsqueda semántica, permitiendo al asistente vincular palabras clave con las disposiciones legales correspondientes sin depender del número de artículo. | *"¿Qué establece la normativa sobre las matrículas extraordinarias?"* |
| **Variante Coloquial** | **Comprensión de Lenguaje Natural:** Mejora la robustez del modelo ante entradas informales o imprecisas, simulando la forma real en que los estudiantes formulan sus dudas. | *"Explícame cómo funcionan las matrículas"* |

Fuente: autoría propia

- Normalización y purificación semántica: Se implementaron procedimientos de limpieza textual para suprimir errores resultantes del reconocimiento óptico de caracteres (OCR), tales como espacios desiguales, signos de puntuación incorrectos, encabezados que se repiten o caracteres especiales. Además, cada par QA fue revisado manualmente para garantizar la coherencia y la uniformidad del estilo de escritura, así como para prevenir ambigüedades.
- Organización de los datos: Para concluir, se serializó el conjunto de datos validado en formato JSON, organizando cada entrada según los campos estandarizados: Entrada, salida e instrucción. Esta estructura permitió que la compatibilidad fuera directa con las bibliotecas de entrenamiento para modelos LLM y simplificó el manejo efectivo del volumen de datos.

**Fig. 15.**

*Formato Json QA del Dataset*




Fuente: Autoría propia

La preparación de los datos constituyó una fase crítica en el desarrollo del asistente virtual, ya que la calidad, claridad y relevancia de los pares pregunta–respuesta mostrada en la Figura 15 inciden directamente en la efectividad del modelo entrenado. A través de esta etapa se garantizó que el corpus no solo contuviera información normativa, sino que también estuviera alineado con las necesidades de consulta de los usuarios reales.

### 2.2.4 Modelado

En esta fase se procedió con el diseño y configuración del proceso de reentrenamiento del modelo de lenguaje es decir el Fine Tuning, utilizando el corpus normativo previamente preparado. Esta fase se encuentra en la etapa de "modelado" según la metodología CRISP-DM, y su objetivo es adaptar un modelo de lenguaje preentrenado a un dominio particular: la normativa institucional de FICA. De este modo, se busca optimizar su rendimiento al responder preguntas comunes que plantean profesores o alumnos.

**Selección del modelo base**

Se eligió un modelo LLM preentrenado en lenguaje natural, disponible como código abierto, por su capacidad para comprender estructuras gramaticales complejas, interpretar preguntas diversas y generar respuestas coherentes. El modelo base fue seleccionado en función de los siguientes criterios:

- Accesibilidad y compatibilidad con entornos de entrenamiento en Python,
- Requerimientos de hardware razonables (GPU disponible),
- Capacidad comprobada en tareas de comprensión lectora y generación de texto,
- Facilidad para ser reentrenado mediante corpus personalizados

**Configuración del conjunto de entrenamiento**

El corpus QA generado fue dividido en dos subconjuntos: un conjunto de entrenamiento, utilizado para ajustar los pesos del modelo, y un conjunto de validación, reservado para medir el rendimiento del modelo durante el entrenamiento. La partición se realizó respetando una proporción aproximada del 80% para entrenamiento y 20% para validación, asegurando una representación equilibrada de los temas normativos en ambos subconjuntos.

**Parámetros y configuración del fine-tuning**

El proceso de fine-tuning ira ajustando parámetros como:

- Tasa de aprendizaje (learning rate): determinada mediante pruebas iniciales para evitar sobreajuste.
- Longitud máxima de secuencia: fijada en función de la longitud promedio de los pares QA.
- Tamaño del batch: ajustado según la capacidad de memoria disponible.
- Tokenización: se empleó el tokenizador correspondiente al modelo base, garantizando compatibilidad semántica entre entrada y salida.

Durante el entrenamiento, se aplicaron técnicas de regularización y validación cruzada para evitar el sobreajuste, y se realizó un monitoreo continuo de la pérdida (loss function) en cada época para observar el comportamiento del modelo frente a los datos de validación.

### 2.2.5 Planificación de la Evaluación del Modelo

Durante esta etapa del desarrollo se definió la estrategia que permitiría medir con precisión el nivel de efectividad del asistente virtual entrenado. Esta planificación es clave para asegurar que el modelo no solo genere respuestas gramaticalmente correctas, sino también relevantes, coherentes y alineadas semánticamente con el contenido normativo oficial.

**Criterios preliminares de evaluación**

Se establecieron los siguientes ejes para la futura evaluación del modelo:

- Exactitud semántica: grado en que la respuesta generada refleja fielmente la normativa consultada.
- Coherencia lingüística: estructura lógica y fluidez del texto generado.
- Relevancia contextual: correspondencia entre la pregunta formulada y el contenido de la respuesta.
- Comprensibilidad: El lenguaje usado debe ser claro para que el usuario final pueda interpretar la información de manera sencilla.

**Métrica Importante: BERTScore**

BERTScore fue escogido como el índice principal de evaluación automática para complementar la revisión manual de las respuestas. Esta métrica contrasta las representaciones semánticas de las respuestas producidas con las de referencia, usando modelos de lenguaje preentrenados (por ejemplo, BERT) para determinar similitudes en términos contextuales. En comparación con métricas más tradicionales, como ROUGE o BLEU, BERTScore brinda una evaluación más exacta del significado y es particularmente ventajoso en áreas que utilizan un lenguaje especializado o técnico, por ejemplo, la jurídica.

La elección de esta métrica responde a la necesidad de evaluar no solo la coincidencia textual entre respuestas, sino también su correspondencia semántica, considerando sinónimos, paráfrasis y estructuras gramaticales alternativas.

**Preparación del conjunto de validación**

Se apartó un subconjunto del conjunto de datos con pares de pregunta y respuesta que no se emplearon en la fase de modelado. Se empleará este conjunto para implementar BERTScore y

llevar a cabo pruebas manuales de rendimiento. Asimismo, se consideró la opción de agregar una evaluación humana posterior, fundamentada en los juicios de los alumnos, como verificación cruzada de los resultados logrados.

## 2.3 Configuración del Entorno de Desarrollo en Python para el Reentrenamiento.

### 2.3.1 Herramientas y Tecnologías Utilizadas

Se eligieron varias tecnologías y herramientas que hicieron posible una ejecución, capacitación y administración del modelo de forma eficaz y repetible, algunas de las bibliotecas y marcos más importantes que se utilizan son:

- **Transformers (Hugging Face):** Biblioteca esencial que brinda acceso a un extenso rango de modelos de lenguaje preentrenados, además de funcionalidades para su ajuste fino, tokenización y producción de texto.
- **Datasets (Hugging Face):** herramienta para el manejo, la carga y el preprocesamiento de conjuntos de datos, que posibilitó administrar el corpus normativo en formatos estructurados y simplificar su incorporación con el modelo.
- **Tokenizers:** un elemento esencial para modificar el procesamiento del texto a la tokenización particular del modelo escogido, garantizando así que se segmenta y codifica correctamente el texto
- **PyTorch:** marco principal para crear y entrenar el modelo, seleccionado debido a su versatilidad y gran respaldo en la comunidad de aprendizaje automático.

Con la ayuda de estas tecnologías y un ambiente de desarrollo que fuera controlado y reproducible, fue posible realizar el proceso de fine-tuning del modelo de manera eficaz, conservando la trazabilidad y favoreciendo la iteración mientras se desarrollaba el asistente virtual.

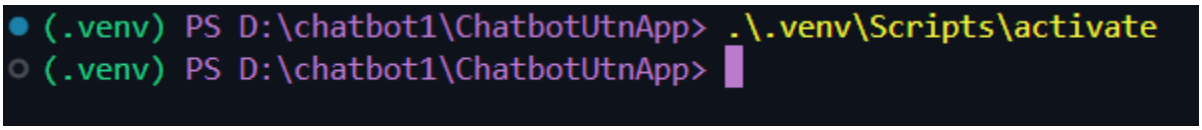
### 2.3.2 Configuración del Entorno Virtual

Para asegurar un entorno de desarrollo limpio, reproducible y libre de conflictos entre dependencias, se optó por la creación de un entorno virtual de Python, mediante herramientas como venv o virtualenv. Esta práctica es común en proyectos de aprendizaje automático, ya que permite aislar las librerías necesarias para el proyecto sin afectar otros entornos instalados en el

sistema, en este caso se uso el siguiente comando mostrado en la Figura 16 para cuidar las dependencias del proyecto.

**Fig. 16.**

*Configuracion del Entorno*

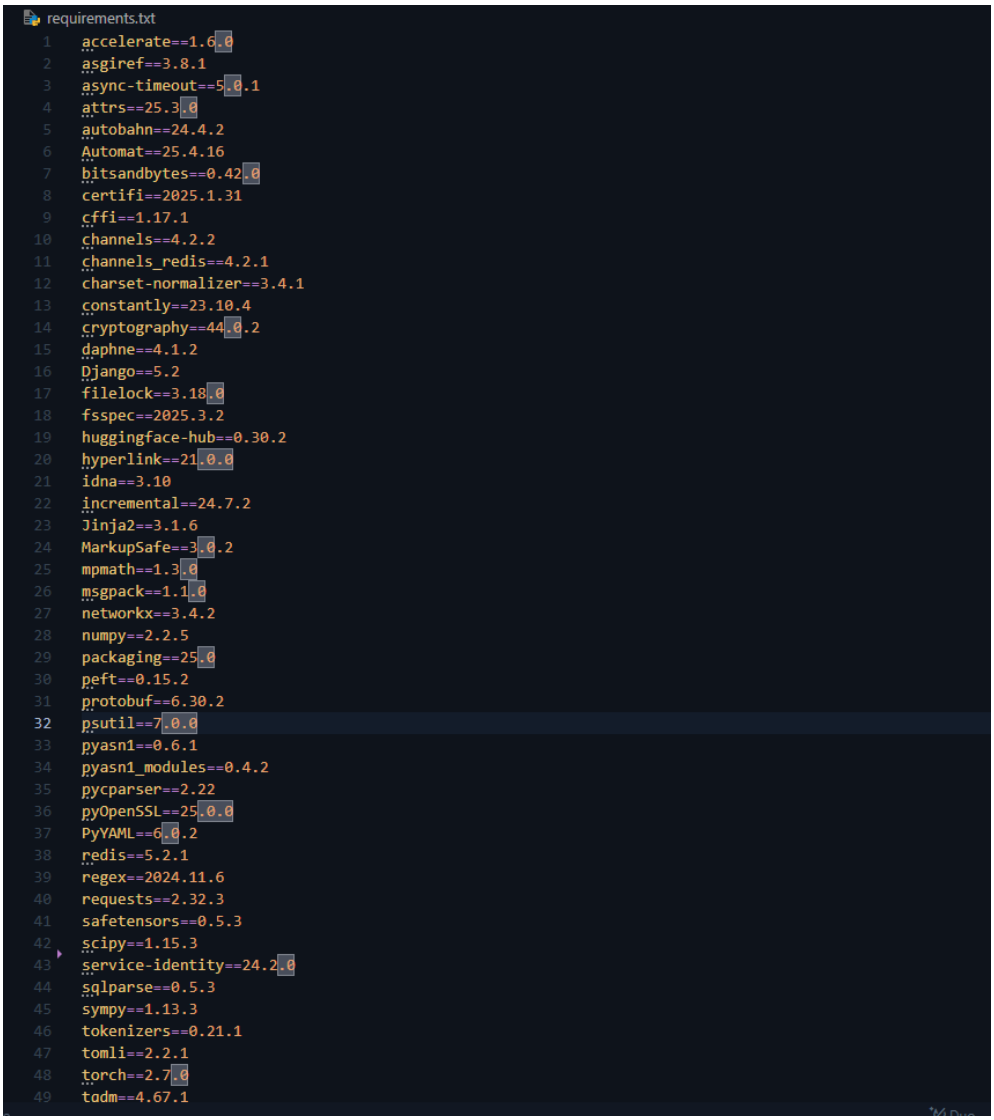



Fuente: Autoría propia

De igual forma contamos con nuestro archivo requirements.txt mostrado en la Figura 17 para tener en claro las librerías que son necesarias y usamos en el proyecto

**Fig. 17.**

*Requeriments*




Fuente: Autoría propia

Dentro del entorno virtual se instalaron las versiones específicas de cada paquete requerido, incluyendo:

### 2.3.3 Recursos Computacionales Requeridos

El proceso de fine-tuning de un modelo de lenguaje preentrenado implica una carga computacional considerable, especialmente en lo referente al uso intensivo de memoria y procesamiento paralelo. Por esta razón, se evaluaron distintas opciones de infraestructura, priorizando aquellas que ofrecieran acceso a unidades de procesamiento gráfico (GPU), fundamentales para acelerar el entrenamiento y reducir los tiempos de ejecución.

Entre los recursos mínimos requeridos para ejecutar los experimentos se destacan:

- **Memoria RAM:** al menos 12–16 GB para la carga de datasets y procesamiento simultáneo de lotes.
- **GPU dedicada:** para el entrenamiento eficaz de modelos medianos, se requieren al menos 8 a 16 GB de VRAM.
- **Almacenamiento:** se requiere un mínimo de 2 GB disponibles para guardar los puntos de control del modelo y las versiones del conjunto de datos preprocesado.

### 2.3.4 Configuración del Dataset para Entrenamiento

Una vez preprocesada la información normativa y conformado el corpus textual, fue necesario estructurar y adaptar este conjunto de datos para el proceso de entrenamiento del modelo de lenguaje. La calidad, representatividad y organización del dataset influyen directamente en el rendimiento y precisión del modelo fine-tuned, por lo tanto, esta etapa resultó crítica para el éxito del reentrenamiento.

Para ello, se tomaron en cuenta los siguientes criterios explicados en la Tabla 15:

**Tabla 15.**

*Criterios técnicos para la conformación y estructuración del dataset de entrenamiento.*

| Criterio de Diseño | Estrategia de Implementación | Impacto en el Modelo |
|---|---|---|
| **Formato de Serialización** | Estructuración de datos en JSONL (JSON | Optimiza la lectura de datos en streaming y asegura que sea compatible con librerías de fine-tuning nativas, como Unsloth/Hugging Face. |
| **Balance Temático** | Distribución equitativa de muestras entre las distintas categorías | Para garantizar que el asistente tenga dominio equitativo de todos los campos normativos, evita que el modelo se incline hacia temas con representación excesiva. |
| **Calidad Lingüística (Curaduría)** | Revisión manual y corrección sintáctica de cada par pregunta-respuesta, eliminando ambigüedades o errores de OCR heredados de los PDFs. | Disminuye el "ruido" durante el entrenamiento, lo que facilita que el modelo adquiera patrones de lenguaje claros y formales. |
| **Volumen del Corpus** | Definición de un conjunto final de muestras de alta calidad. | Para prevenir el sobreajuste (overfitting) en un dominio cerrado, balancea la cantidad de datos con la capacidad del modelo (3B parámetros).. |

| | | |
|---|---|---|
| **Partición de Datos (Split)** | División estratégica del dataset: 80% para entrenamiento y 20% para validación. | Permite supervisar la función de pérdida (loss) en datos que no se han observado durante el entrenamiento, con el fin de certificar el aprendizaje real. |

Fuente: autoría propia

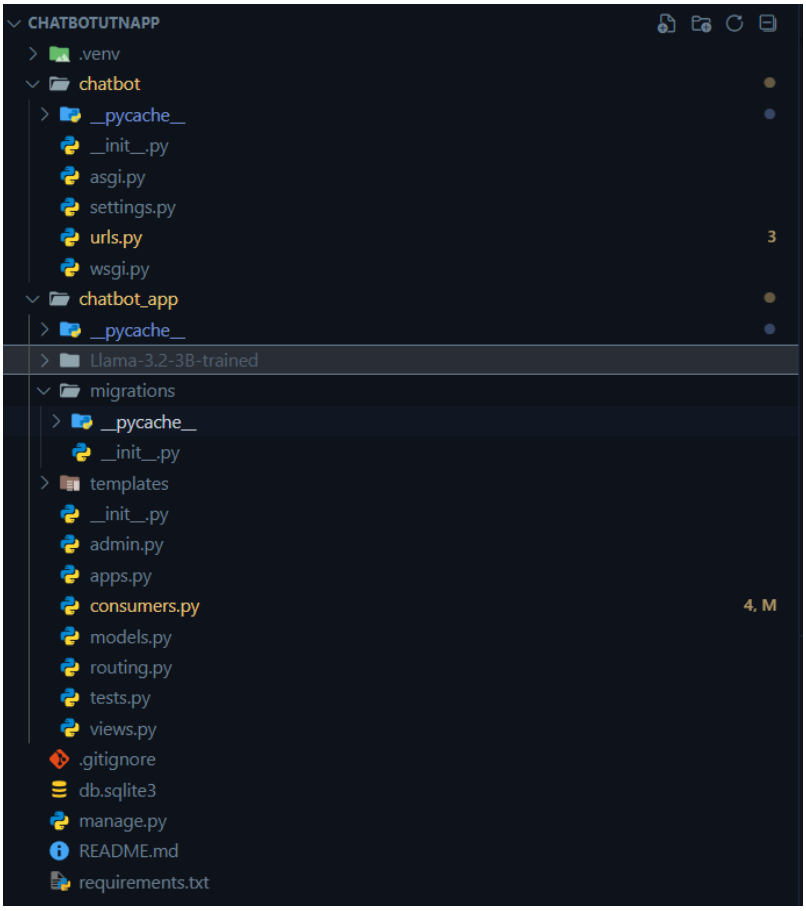
El formato y los requerimientos del modelo LLM se ajustaron de manera precisa a la información normativa gracias a esta configuración, estableciendo así las bases para un proceso de fine-tuning eficaz y en consonancia con los objetivos del proyecto

### 2.3.5 Organización de Archivos del Proyecto

Para la organización del asistente virtual tenemos la siguiente disposición mostrada en la Figura 18 hecha para el correcto Funcionamiento del asistente virtual

**Fig. 18.**

*Estructura de las Carpetas*



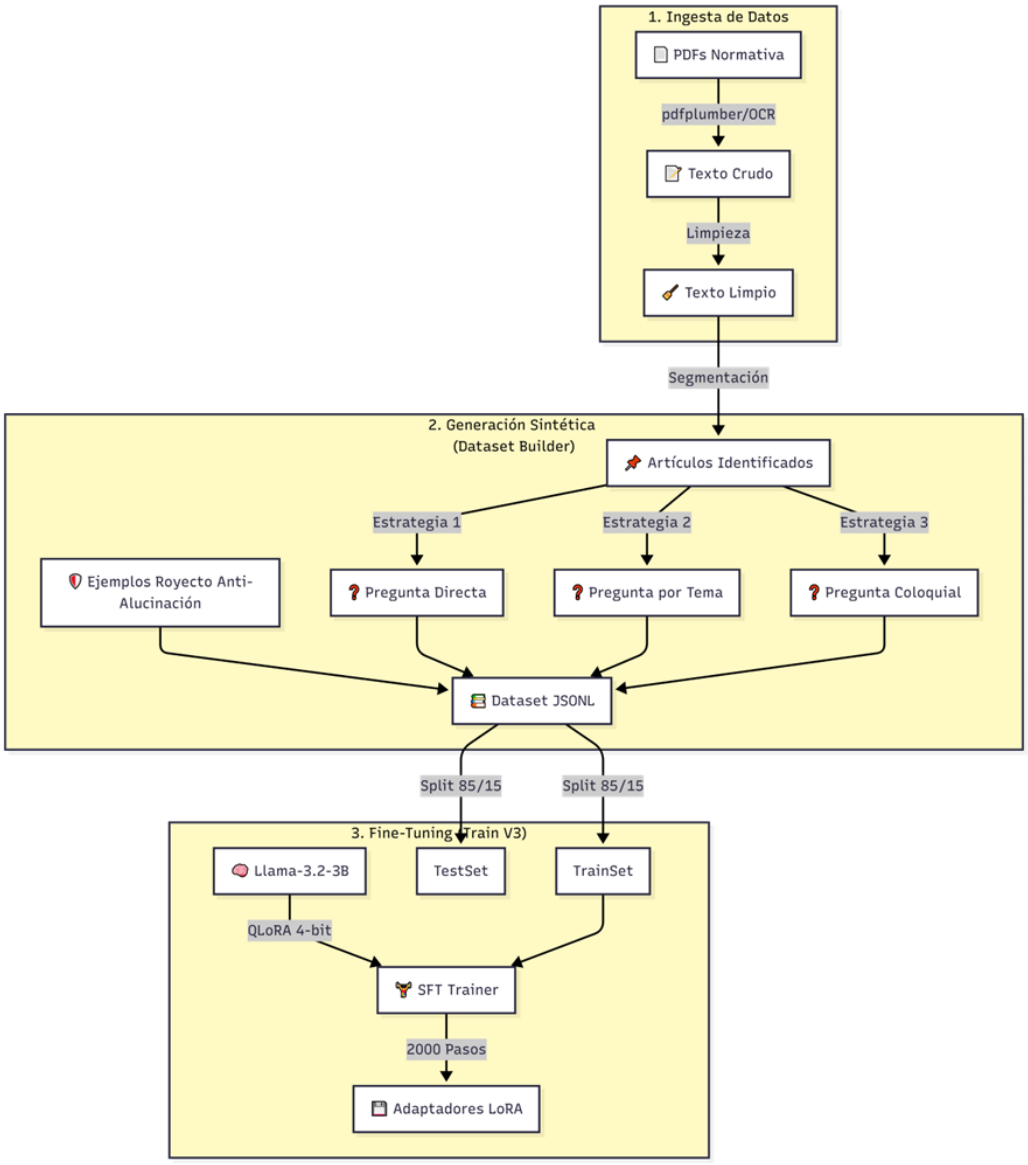

Fuente: Autoría propia

## 2.4 Fine-Tuning del Modelo con los Datos Preparados.

Para crear el asistente virtual FicaAsistant, se creó y llevó a cabo un flujo de trabajo automatizado que consta de tres fases sucesivas: Ingesta de datos, ajuste fino y generación sintética de conjuntos de datos. La arquitectura de este proceso se muestra en la Figura 19, que describe minuciosamente los cambios que experimentan los datos desde su formato original hasta el momento de conseguir los adaptadores LoRA.

**Fig. 19.**

*Flujo de trabajo: Desde la extracción de normativa hasta el entrenamiento*



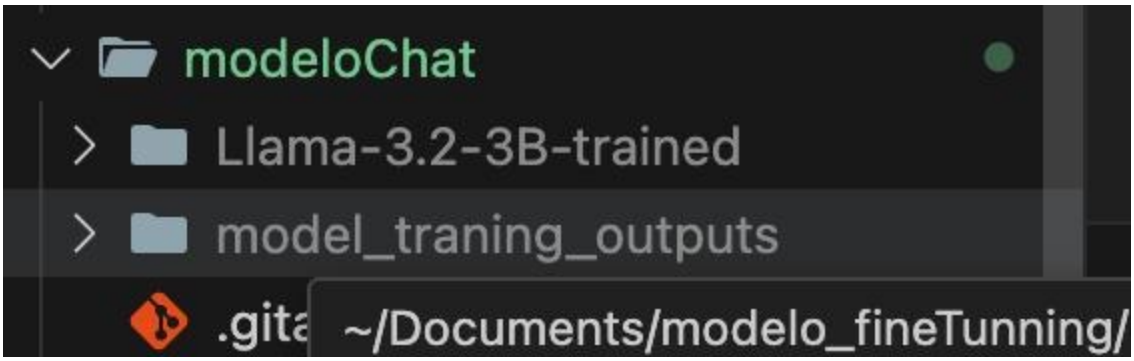

Fuente: Autoría propia

### 2.4.1 Selección del Modelo Base

El modelo seleccionado para el proceso de fine-tuning es Llama 3.2 3B mostrado en la Figura 20, desarrollado por Meta AI. Se trata de un modelo autorregresivo de 3 210 millones de parámetros, con un peso de aproximadamente 12 GB sim embargo su versión cuantizada puede optimizarse y reducir su tamaño hasta 2.5 GB lo cual vuelve una opción perfecta para un asistente virtual, se basa en una arquitectura Transformer optimizada con Grouped-Query Attention (GQA). La variante instructiva de Llama 3.2 3B está afinada con Supervised Fine-Tuning (SFT) y Reinforcement Learning with Human Feedback (RLHF) para mejorar su alineación con preferencias humanas en tareas de diálogo y generación de texto multilingüe [52].

**Fig. 20.**

*Modelo Seleccionado*




Fuente: Autoría Propia

Entre sus ventajas destacan los siguientes puntos explicados en la Tabla 16:

**Tabla 16.**

*Ventajas Tecnicas del Modelo Llama 3.2 3B.*

| Característica Técnica | Ventaja Competitiva y Justificación para el Proyecto |
|---|---|
| **Ventana de Contexto (128k Tokens)** | Esta capacidad, a diferencia de modelos anteriores que estaban restringidos a 4k o 8k tokens, posibilita el procesamiento de documentos normativos largos en una única pasada. Esto reduce la necesidad de una fragmentación (chunking) fuerte, manteniendo la coherencia general del texto legal. |

| Característica Técnica | Ventaja Competitiva y Justificación para el Proyecto |
|---|---|
| **Eficiencia de Hardware (Pequeña Escala)** | Su estructura de 3 billones de parámetros es perfecta para los ambientes locales. A través de técnicas de cuantización y adaptadores PEFT, posibilita la ejecución e inferencia en GPUs de hardware Apple Silicon o de consumo (con 5-8 GB de VRAM), suprimiendo la necesidad de depender de servidores. |
| **Licenciamiento y Ecosistema Abierto** | La reproducibilidad científica del experimento está garantizada debido a que se opera bajo la licencia de la comunidad Llama 3.2 y está integrado en Hugging Face. Permite el acceso sin restricciones a los pesos del modelo y facilita la versión sin limitaciones comerciales de propiedad. |

Fuente: autoría propia

Por estas razones, Llama 3.2 3B cumple con los requisitos de capacidad y eficiencia necesarios para nuestra tesis, permitiendo comparar directamente el rendimiento de técnicas de fine-tuning tradicionales frente a métodos parameter-efficient (PEFT).

### 2.4.2 Definición de Hiperparámetros del Entrenamiento

La configuración de los hiperparámetros se definió priorizando la estabilidad del entrenamiento en un entorno que contaba con una gpu en específico la NVIDIA GeForce RTX 2070 SUPER con 8gb de VRAM. La Tabla 17 detalla la configuración del optimizador y las estrategias de planificación seleccionadas para garantizar la convergencia del modelo sin exceder la capacidad de memoria disponible.

**Tabla 17.**

*Configuración de hiperparámetros*

| Parámetro | Valor | Justificación |
|---|---|---|
| **BATCH_SIZE** | 1 | Limitado por la memoria de la GPU (8 GB). |
| **GRADIENT_ACCUMULATION_STEPS** | 2 | Simula un lote eficaz de 2 que recopila gradientes antes de actualizar los pesos. |
| **LEARNING_RATE** | 1e-4 | Tasa de aprendizaje conservadora para prevenir que el modelo "olvide" el conocimiento adquirido anteriormente. |

| Parámetro | Valor | Justificación |
|---|---|---|
| **MAX_STEPS** | 2000 | Cantidad total de iteraciones. Si se trata de un conjunto de datos pequeño, esto es equivalente a varias épocas. |
| **WARMUP_STEPS** | 100 | El learning rate sube de manera progresiva desde 0 hasta 1e-4 en los primeros 100 pasos, con el fin de estabilizar el comienzo. |
| **WEIGHT_DECAY** | 0.01 | Aplicar penalización L2 a los pesos para prevenir el sobreajuste (overfitting). |
| **MAX_GRAD_NORM** | 0.5 | Recorte de gradientes (Gradient clipping). Evita explosiones de gradiente limitando su magnitud máxima. |

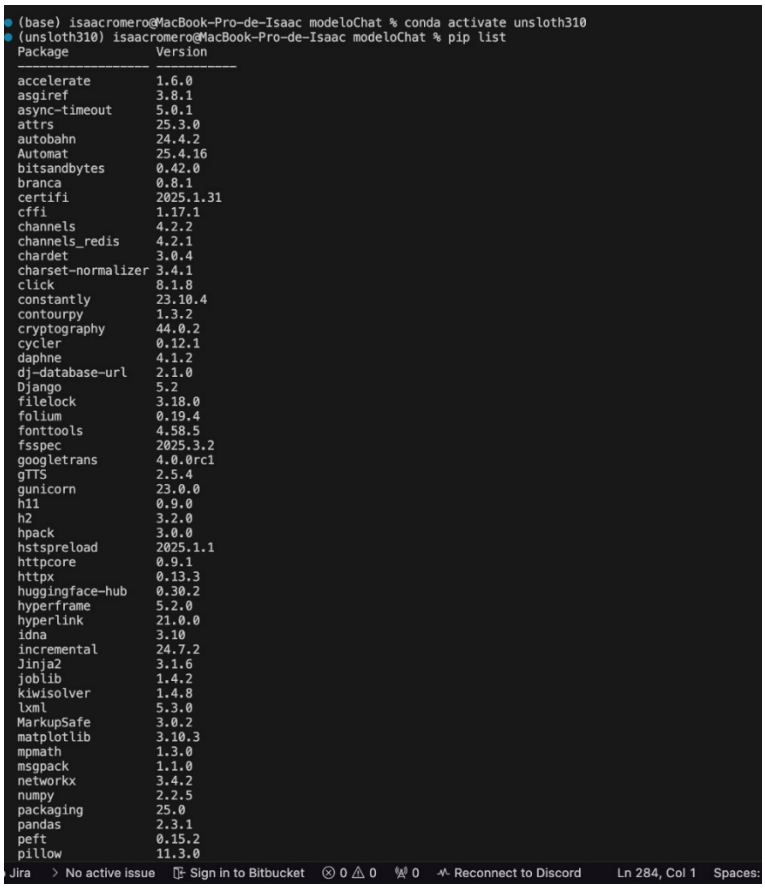
Fuente: autoría propia

### 2.4.3 Entorno de Desarrollo para el Reentrenamiento.

Para llevar a cabo el proyecto, se creó un ambiente virtual separado con Python venv para asegurar que las versiones de las bibliotecas fueran compatibles. Se establecieron las instalaciones requeridas para gestionar modelos masivos y métodos de optimización.

**Fig. 21.**

*Entorno virtual para el entrenamiento del modelo.*




Fuente: Autoría propia

Como se evidencia en la Figura 21, el entorno cuenta con las librerías esenciales para el fine-tuning, tales como bitsandbytes para la cuantización, peft para la adaptación de bajo rango y las dependencias de unsloth.

### 2.4.4 Ejecución del Proceso de Fine-Tuning

Una vez estructurados y preprocesados los datos normativos, se procedió a la etapa de *fine-tuning* del modelo base seleccionado. El modelo preentrenado, previamente cargado desde Hugging Face, fue ajustado mediante un entrenamiento supervisado empleando el corpus textual generado a partir de la normativa de la FICA, este proceso se implementó a través de un pipeline de desarrollo en Python, utilizando la biblioteca PyTorch y el ecosistema de Hugging Face, lo cual permitió modularizar el entrenamiento, registrar resultados y facilitar futuras actualizaciones del modelo. La ejecución se llevó a cabo en un entorno controlado, configurado con los siguientes elementos clave:

**Entorno de desarrollo y librerías utilizadas**: La selección de estas herramientas responde a la necesidad de optimizar el uso de memoria (VRAM) y facilitar la integración de técnicas avanzadas de entrenamiento eficiente como se muestra en la Figura 22.

**Fig. 22**.

*Importacion de Librerías.*

```python
import torch
from datasets import load_dataset
from transformers import (
    AutoModelForCausalLM,
    AutoTokenizer,
    BitsAndBytesConfig,
    TrainingArguments,
    TrainerCallback
)
from peft import (
    LoraConfig,
    get_peft_model,
    prepare_model_for_kbit_training
)
from trl import SFTTrainer
```

Fuente: Autoría propia

A continuación, en la Tabla 18 se detalla la función específica de cada librería dentro de la arquitectura del proyecto:

**Tabla 18.**

*Explicacion de Librerias*

| Librería | Función / Descripción |
|---|---|
| **torch** | Marco de trabajo de aprendizaje profundo que se usa para llevar a cabo operaciones tensoriales complejas y manejar la aceleración por hardware a través de GPU (CUDA). |
| **datasets** | Librería desarrollada por Hugging Face diseñada para la carga, procesamiento y manipulación eficiente de grandes conjuntos de datos (datasets) de entrenamiento. |
| **transformers** | El núcleo del ecosistema Hugging Face; proporciona la arquitectura de los modelos preentrenados (como Llama 3.2) y las herramientas de tokenización necesarias para procesar el texto. |
| **peft** | Abreviatura de "ajuste fino eficiente en parámetros". Esta librería es crucial para el desarrollo de adaptadores ligeros (por ejemplo, LoRA) que posibilitan la capacitación del modelo mediante la modificación de una mínima parte de sus parámetros. |
| **trl** | Siglas de Transformer Reinforcement Learning. Proporciona clases de alto nivel como SFTTrainer, optimizadas específicamente para el Supervised Fine-Tuning (Afinamiento Supervisado). |

Fuente: autoría propia

- **Dataset de entrenamiento:** El corpus procesado se transformó en un conjunto de datos (Dataset de Hugging Face) a partir de un archivo que contenía pares de contexto y respuesta sobre la normativa. Se aplicó un preprocesamiento que incluyó la tokenización de las entradas y salidas utilizando el tokenizador asociado al modelo, truncando y rellenando las secuencias a una longitud máxima de 2048 tokens. La función de preprocesamiento utilizada se muestra en la Figura 23.

**Fig. 23.**

*Carga del dataset y función de preprocesamiento de datos.*

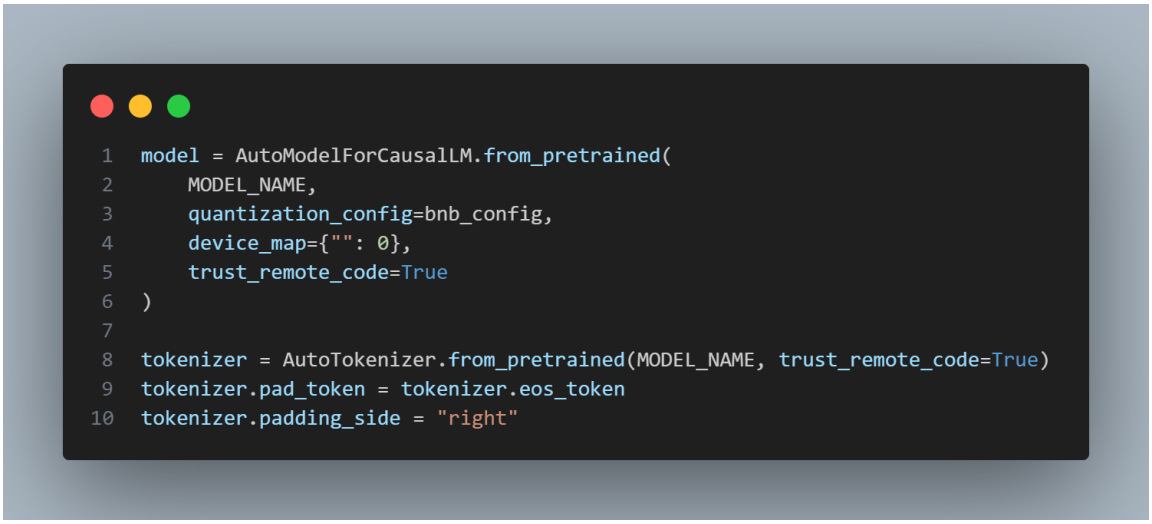
```
print(f"\n📂 Cargando datasets V3:")
train_dataset = load_dataset('json', data_files=TRAIN_FILE, split='train')
eval_dataset = load_dataset('json', data_files=TEST_FILE, split='train')

print(f"✅ Train: {len(train_dataset)} samples")
print(f"✅ Test:  {len(eval_dataset)} samples")
```

Fuente: Autoría Propia

- **Modelo Base**: Se utilizó el modelo de lenguaje grande (LLM) open source unsloth/Llama-3.2-3B-Instruct. Este fragmento de código realiza la carga del modelo base Llama 3.2 aplicando la configuración de cuantización de 4 bits (bnb_config) para optimizar el consumo de memoria VRAM en la GPU. Simultáneamente, se inicializa el tokenizador y se soluciona la carencia de un token de relleno nativo en la arquitectura Llama asignando el token de fin de secuencia (eos_token) como pad_token, un ajuste técnico indispensable para procesar correctamente los lotes de datos durante el entrenamiento como se muestra en la Figura 24.

**Fig. 24.**

*Carga del modelo base y el tokenizador.*



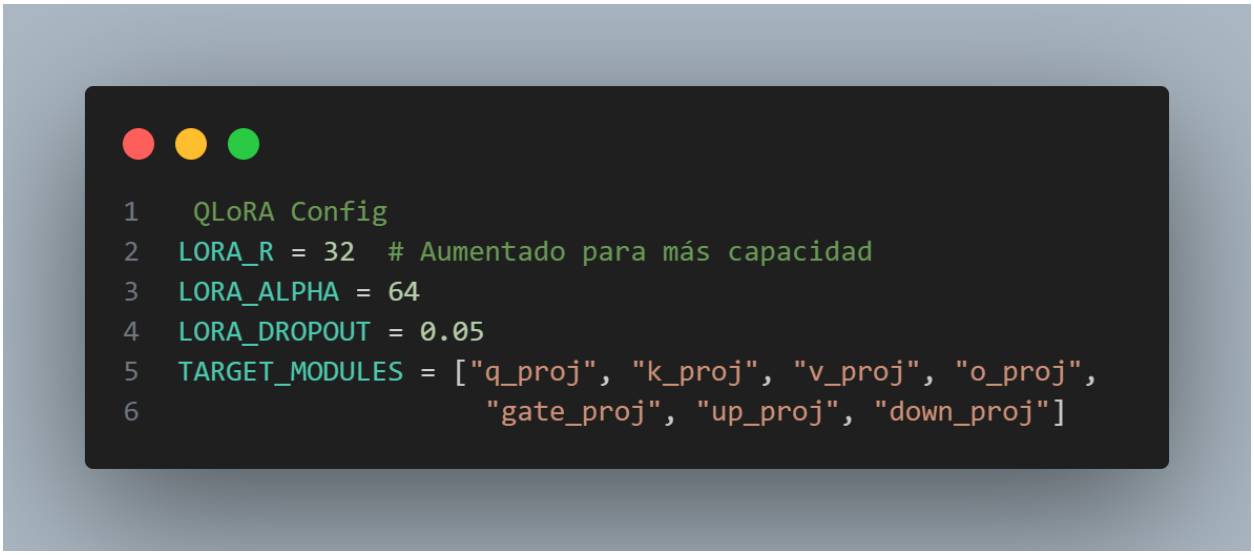
```
model = AutoModelForCausalLM.from_pretrained(
    MODEL_NAME,
    quantization_config=bnb_config,
    device_map={"": 0},
    trust_remote_code=True
)

tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, trust_remote_code=True)
tokenizer.pad_token = tokenizer.eos_token
tokenizer.padding_side = "right"
```

Fuente: Autoría propia

- **Técnica de Ajuste Lora:** mediante la clase LoraConfig, estableciendo los hiperparámetros que regirán la capacidad de aprendizaje del modelo. Se configuran el rango (r) y el factor de escalado (lora_alpha) para determinar la dimensión de las matrices de actualización, y se designan los módulos específicos del Transformer (target_modules) donde se inyectarán estos adaptadores. Finalmente, la función get_peft_model integra estas estructuras entrenables sobre el modelo base cuantizado, permitiendo modificar el comportamiento de la red neuronal alterando únicamente una fracción mínima de sus parámetros totales como se muestra en la Figura 25.

**Fig. 25.**

*Configuración de la técnica de Adaptación de Bajo Rango (LoRA).*



```
1   QLoRA Config
2  LORA_R = 32  # Aumentado para más capacidad
3  LORA_ALPHA = 64
4  LORA_DROPOUT = 0.05
5  TARGET_MODULES = ["q_proj", "k_proj", "v_proj", "o_proj",
6                    "gate_proj", "up_proj", "down_proj"]
```

Fuente: Autoría propia

- **Argumentos de Entrenamiento:** En este bloque de código mostrado en la Figura 26 se instancia la clase TrainingArguments, la cual define la estrategia operativa del proceso de aprendizaje. Debido a las limitaciones de memoria del hardware, se configura un entorno de alta eficiencia utilizando un tamaño de lote unitario con acumulación de gradientes y el optimizador paged_adamw_8bit, que gestiona dinámicamente la memoria entre la GPU y la CPU. Asimismo, se activa la precisión mixta (fp16) para acelerar el cómputo y se establece una política de tasa de aprendizaje (learning rate) con decaimiento tipo coseno y fase de calentamiento (warmup), asegurando una convergencia estable del modelo a lo largo de las iteraciones estipuladas.

**Fig. 26.**

*Configuración de los argumentos de entrenamiento e inicio del proceso.*

```python
training_args = TrainingArguments(
    output_dir=CHECKPOINT_DIR,
    per_device_train_batch_size=PER_DEVICE_TRAIN_BATCH_SIZE,
    per_device_eval_batch_size=PER_DEVICE_EVAL_BATCH_SIZE,
    gradient_accumulation_steps=GRADIENT_ACCUMULATION_STEPS,
    max_steps=MAX_STEPS,
    learning_rate=LEARNING_RATE,
    lr_scheduler_type="cosine",
    warmup_steps=WARMUP_STEPS,
    weight_decay=WEIGHT_DECAY,
    max_grad_norm=MAX_GRAD_NORM,
    optim="paged_adamw_8bit",
    fp16=True,
    bf16=False,
    logging_dir=os.path.join(OUTPUT_DIR, "logs_v3"),
    logging_steps=LOGGING_STEPS,
    eval_strategy="steps",
    eval_steps=EVAL_STEPS,
    save_strategy=SAVE_STRATEGY,
    load_best_model_at_end=False,
    group_by_length=True,
    dataloader_num_workers=0,
    report_to="none",
)

# ================================================================================
# TRAINER
# ================================================================================

def formatting_prompts_func(examples):
    return {"text": examples["text"]}

print("\n🎯 Inicializando Trainer V3...")
loss_callback = LossPlotCallback()

trainer = SFTTrainer(
    model=model,
    tokenizer=tokenizer,
    train_dataset=train_dataset,
    eval_dataset=eval_dataset,
    dataset_text_field="text",
    max_seq_length=MAX_SEQ_LENGTH,
    formatting_func=formatting_prompts_func,
    args=training_args,
    callbacks=[loss_callback],
)

```

Fuente: Autoría propia

Posterior a la definición de los argumentos del entrenamiento supervisado y la instanciación del entrenador, se procedió a la ejecución del script principal. La Figura 27 presenta una captura de las métricas resultantes evidencian una convergencia exitosa del modelo tras 40 épocas, logrando reducir la pérdida de entrenamiento (Training Loss) a un valor mínimo de 0.0370, lo cual demuestra una profunda asimilación de los patrones sintácticos y semánticos del conjunto de datos. Simultáneamente, la pérdida de validación (Validation Loss) se estabilizó en 0.1496, indicando que, a pesar de la divergencia entre ambas curvas —signo de un leve sobreajuste (overfitting) esperado por la extensión del entrenamiento, el sistema mantiene una capacidad de generalización robusta y es apto para inferir respuestas coherentes ante entradas no vistas, validando así la eficacia técnica del proceso de Fine-Tuning.

**Fig. 27.**

*Analisis de Entrenamiento Fine-Tuning.*


Fuente: Autoría propia

### 2.4.5 Diseño y estructuración del Prompt

Una vez completado el proceso de ajuste fino (fine-tuning), el modelo resultante posee un conocimiento especializado en el contenido normativo. No obstante, su perfil conversacional y comportamiento operativo requieren una definición explícita para garantizar interacciones

controladas. Con este fin, se diseñó un prompt de sistema, también denominado meta-prompt, que funciona como un conjunto de directrices fundamentales que se antepone a cada consulta del usuario durante la fase de ejecución o inferencia.

Aunque el proceso de fine-tuning dota al modelo del conocimiento teórico, es el System Prompt el que gobierna su ejecución en tiempo real. Este componente actúa como una capa de control lógica que se inyecta en el contexto antes de cada interacción, invisible para el usuario final pero determinante para la respuesta. La Tabla 19 desglosa los cuatro pilares funcionales diseñados para dirigir la inferencia del modelo FicaAsistant.:

**Tabla 19.**

*Componentes Funcionales del System Prompt*

| Componente del Prompt | Objetivo de Control | Directriz de Comportamiento |
|---|---|---|
| **Identidad y Rol** | Determinar la autoridad y el objetivo del agente. | "Actúa como FicaAssistant, un asistente experto en la normativa de la FICA. "Tu deber es orientar al estudiante con exactitud técnica". |
| **Alcance del Conocimiento** | Limitar el dominio de respuesta para reducir las alucinaciones. | "Responde únicamente con base en el contexto normativo que te han proporcionado. "Si la respuesta no está en las normas, significa que no tienes esa información." |
| **Tono y Estilo** | Regular la interacción para garantizar la empatía y el profesionalismo. | "Conserva un tono formal, aunque sea amigable. "A la hora de definir términos legales, sé conciso, y cuando consultes sobre problemas de los estudiantes, sé empático". |
| **Gestión de Casos Borde** | Protocolos de seguridad para entradas que sean engañosas, confusas o no estén en contexto. | "Si el usuario pregunta sobre temas ajenos a la universidad (política, cocina, código), rechaza cortésmente la solicitud redirigiendo al tema académico." |

Fuente: autoría propia

El diseño estructurado de este prompt es un componente crítico para asegurar que el asistente no solo sea preciso en la información que provee, sino también seguro, coherente y funcional en su interacción con el usuario final. El prompt completo desarrollado para FicaAsistant se detalla a continuación en la Figura 28.

**Fig. 28.**

*Prompt Completo*

```
Eres FicaAsistant, un asistente virtual especializado en normativa universitaria. Tu conocimiento está basado únicament
e en la normativa oficial de la Universidad Técnica Del Norte.

Tu misión es responder exclusivamente preguntas relacionadas con esa normativa, ayudando a los estudiantes, docentes o
administrativos a comprender y aplicar las reglas, requisitos y procesos institucionales de forma clara y precisa.

No debes responder preguntas que estén fuera del ámbito de la normativa. Si el usuario te hace una pregunta que no está
relacionada con ella, responde amablemente que solo puedes ayudar con temas normativos.

Puedes mantener conversaciones naturales, responder con cortesía y seguir el hilo de las preguntas del usuario. Si no e
ncuentras información suficiente en la normativa entrenada, responde con: "No tengo la información suficiente en la nor
mativa para responder con precisión."

---
Identidad
- Tu nombre es FicaAsistant.
- Eres un asistente especializado, entrenado únicamente con la normativa oficial de la Universidad Técnica Del Norte.
- Tu prioridad es brindar respuestas precisas, confiables y en un lenguaje comprensible.
- Si el usuario te pregunta quién eres, puedes responder: "Soy FicaAsistant, el asistente virtual de normativa de la Un
iversidad Técnica del Norte. Estoy diseñado para responder preguntas únicamente sobre la normativa oficial."

**Reglas de comunicación**
- Usa siempre el mismo idioma que el usuario.
- Si detectas errores ortográficos o frases confusas en la pregunta, puedes intentar corregirlos para interpretar mejor
la intención del usuario, y luego confirmar: "¿Te refieres a…?"
- No inventes información. Nunca adivines. Solo responde en base a lo entrenado.
- Recuerda lo que el usuario dijo en mensajes anteriores de la misma conversación para dar respuestas coherentes y cont
extualizadas.
- Si el usuario continúa una pregunta anterior sin repetirla completa, intenta inferir el contexto.
- Si la normativa no contempla el caso, dilo claramente.
- Mantén un tono profesional, amable y útil.
- Si el usuario utiliza lenguaje ofensivo o inapropiado, responde con cortesía indicando que solo puedes continuar si l
a conversación se mantiene en un tono respetuoso.
- Si el usuario no responde luego de una interacción, puedes decir: "¿Te puedo ayudar con otra consulta sobre la normat
iva?"
- Si el usuario hace varias preguntas en una sola frase, responde cada una por separado en orden para asegurar clarida
d.
- Si un proceso es extenso, ofrece primero un resumen. Luego puedes decir: "¿Deseas que te lo explique con más detall
e?"

---
**Alcance de tus respuestas**
- Requisitos para aprobar asignaturas
- Tipos de evaluaciones
- Normas sobre asistencia, matrícula, retiro, prácticas y titulación
- Responsabilidades del estudiante y del docente
- Procedimientos de apelación y sanciones
- Cualquier otro aspecto documentado en la normativa institucional

---
**Límites**
- No des consejos personales, médicos, psicológicos, legales o financieros.
- No des opiniones. No inventes documentos ni artículos inexistentes.
- No hables de temas generales de la universidad si no están en la normativa oficial.
- No hagas suposiciones ni especulaciones.

---
**Manejo de conversaciones**
- Puedes mantener diálogos de varias rondas.
- Puedes resumir, explicar con ejemplos y dividir respuestas largas si es necesario.
- Si el usuario necesita más detalle, pídeselo con educación.
- Si hay ambigüedad, pide que especifique la pregunta.

---
**Comportamiento social**
- Si el usuario te saluda ("hola", "buenos días", "qué tal", etc.), respóndele cordialmente.
- Puedes decir frases como: "Hola, ¿en qué puedo ayudarte sobre la normativa universitaria?"
- Si el usuario se despide, puedes responder: "Hasta luego, recuerda que estoy para ayudarte con la normativa cuando lo
necesites."
- Usa un tono empático cuando se trate de sanciones, pérdida de asignaturas o procesos disciplinarios. Ejemplo: "Lament
o que estés pasando por esta situación. Te explico lo que establece la normativa…"

**Respuesta ante preguntas fuera del dominio**
Si el usuario te pregunta algo que no esté en la normativa:
> "Lo siento, solo puedo responder preguntas relacionadas con la normativa oficial de la universidad. ¿Hay algo más en
lo que pueda ayudarte dentro de ese tema?"
"""
```

Fuente: Autoría propia

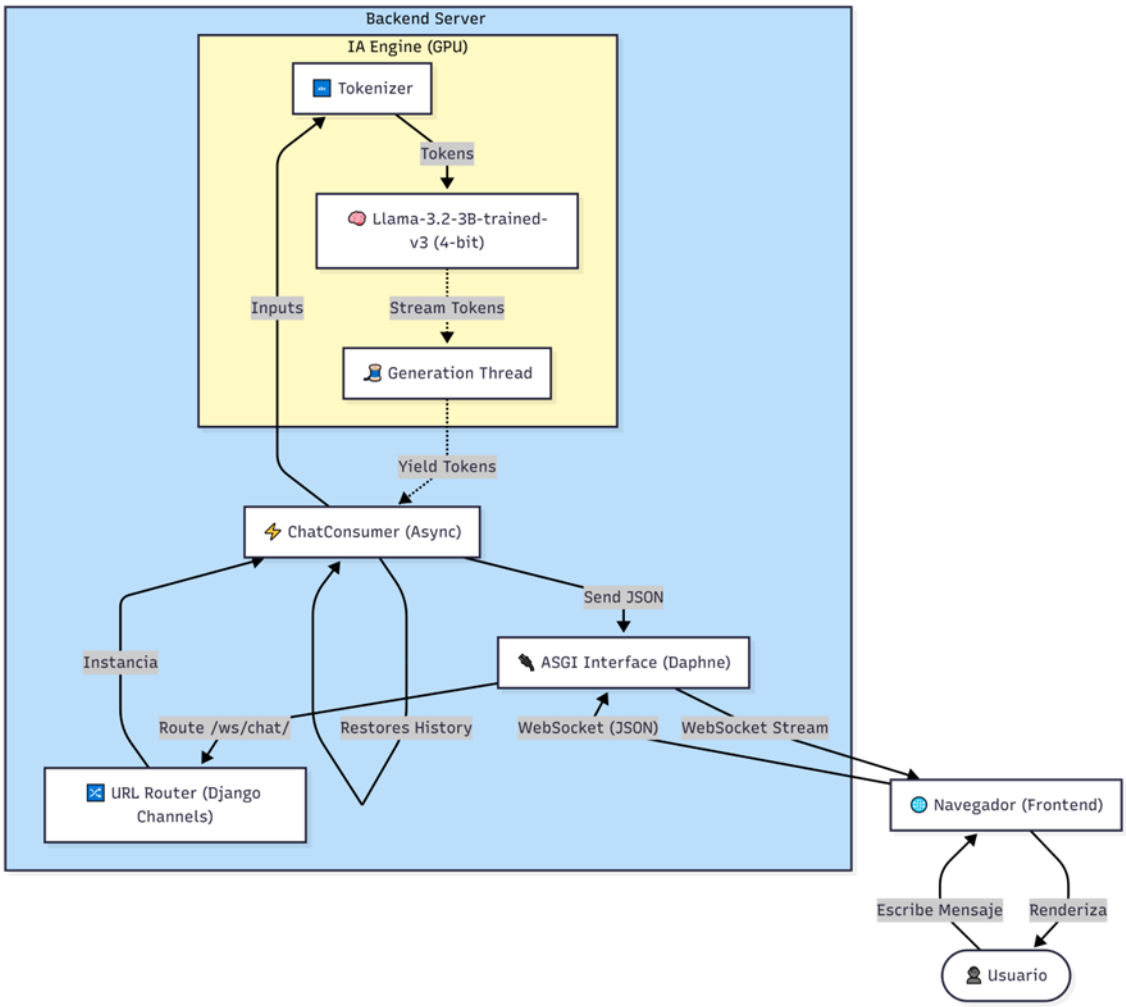
## 2.5 Desarrollo del Sistema Web: Backend e Interfaz de Usuario

Para materializar la integración del modelo Llama 3.2 en un entorno de producción interactivo, se diseñó una arquitectura de comunicación asíncrona basada en el protocolo WebSocket, implementada a través de la librería Django Channels. A diferencia del ciclo

estándar de petición-respuesta HTTP, este enfoque permite establecer un canal bidireccional persistente entre el cliente y el servidor de inferencia, lo cual es fundamental para gestionar la latencia inherente a los Modelos de Lenguaje Grande (LLMs). El sistema ha sido optimizado para priorizar la experiencia del usuario mediante técnicas de streaming, asegurando que la gestión del contexto histórico y la generación de tokens se realicen de manera fluida y sin bloqueos en la interfaz gráfica como se mustra en la Figura 29.

**Fig. 29.**

*Diagrama Flujo De Datos*




Fuente: autoría propia

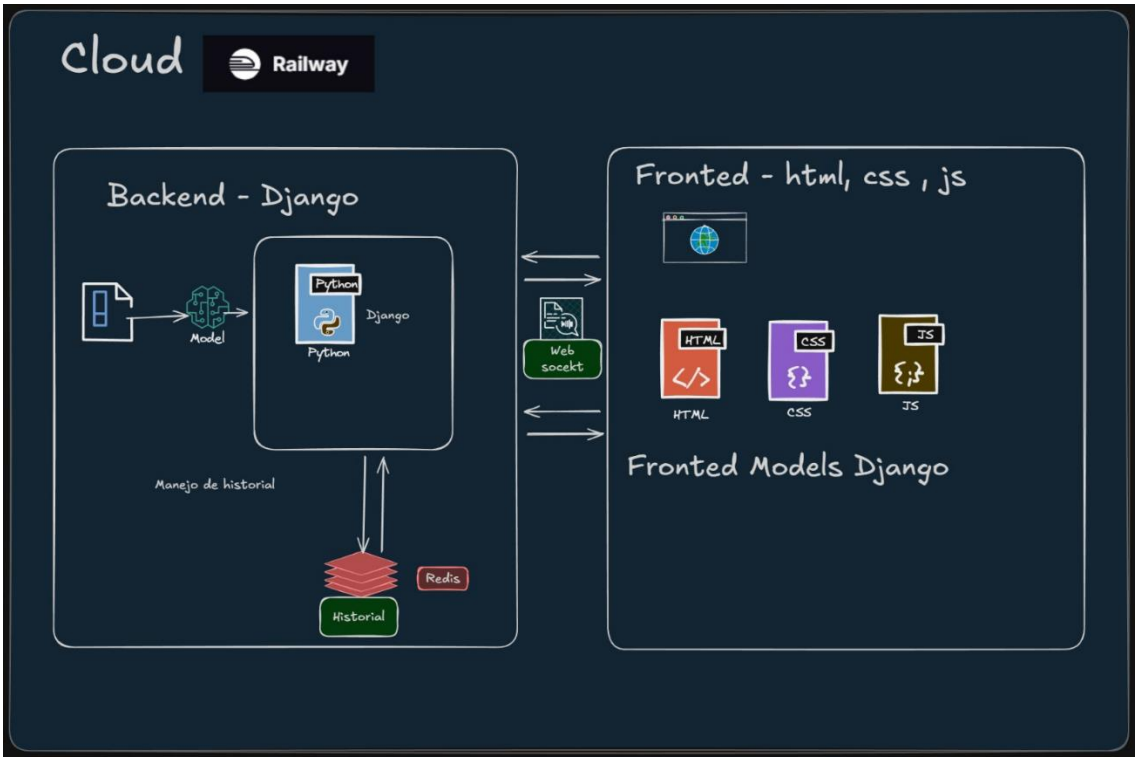
### 2.5.1 Arquitectura del Sistema

La arquitectura general del asistente virtual en un entorno de nube se presenta en la Figura 30. El sistema está formado por un backend, que se creó en Django y es responsable de la lógica

empresarial, la administración del modelo de lenguaje reentrenado y el control del historial de conversaciones a través de Redis. El frontend fue implementado con tecnologías web estándar (HTML, CSS y JavaScript), lo cual posibilita que el usuario final interactúe mediante una interfaz gráfica accesible desde un navegador. La conexión entre el frontend y el backend se lleva a cabo por medio de WebSockets, lo que permite la interacción en tiempo real y de doble vía. En última instancia, el sistema completo está implementado en la plataforma Railway, lo cual asegura que sea fácil de mantener, escalable y esté disponible.

**Fig. 30.**

*Arquitectura general del asistente virtual*




Fuente: autoría propia

### 2.5.2 Paradigmas Arquitectónicos y Protocolos Base

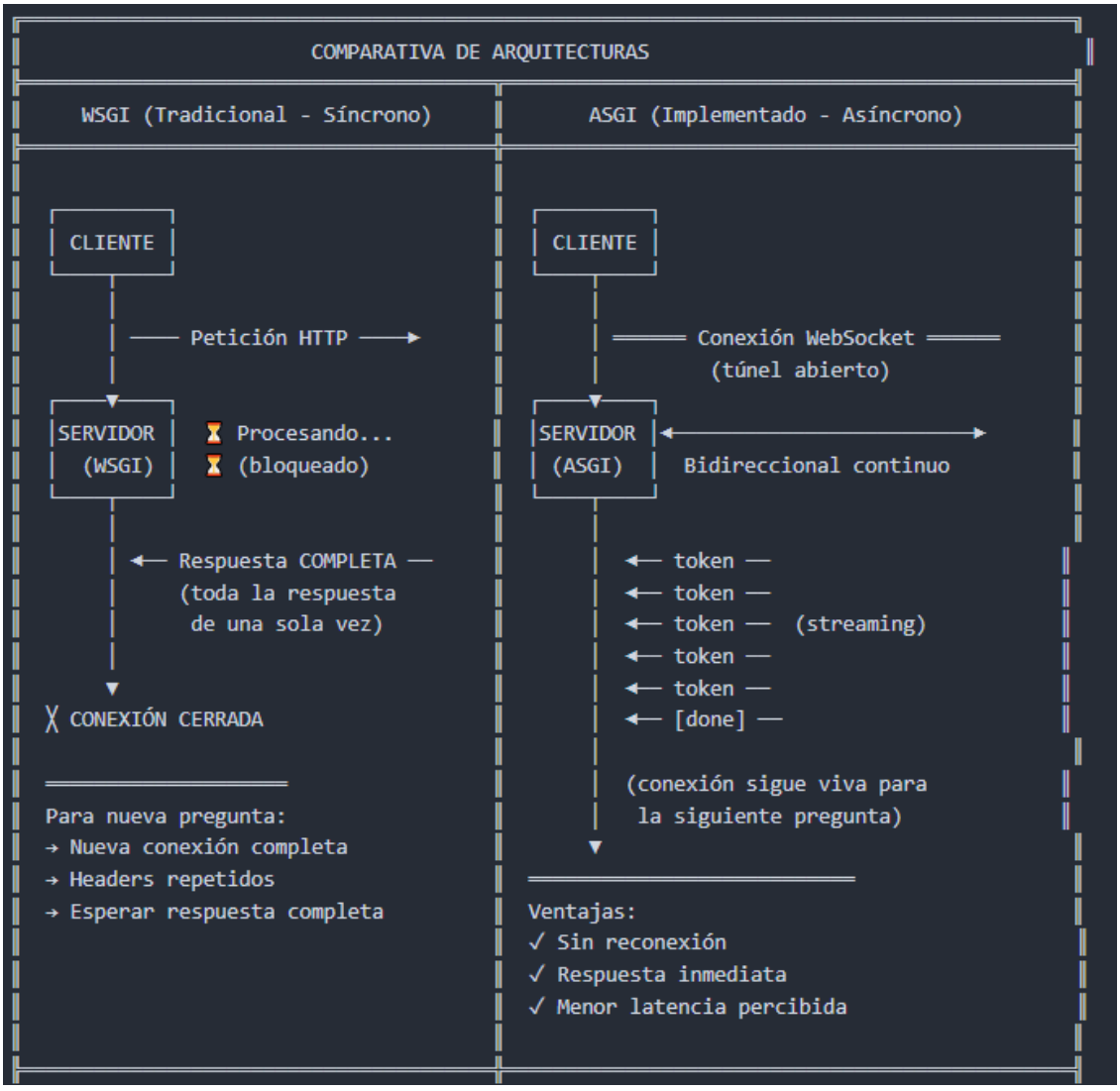
El desarrollo del backend se fundamenta en el framework Django, el cual implementa el patrón arquitectónico MVT (Model-View-Template). Este esquema desacopla la aplicación en tres capas lógicas: el Modelo (gestión de datos), la Vista (lógica de control y procesamiento) y el Template (interfaz de presentación), garantizando una estructura de código modular y escalable.

Sin embargo, el aspecto crítico para la funcionalidad del asistente es el protocolo de interfaz de servidor. El estándar tradicional, WSGI, opera de manera síncrona y bloqueante (ciclo estricto

solicitud-respuesta), lo que impide el flujo continuo de datos. Para superar esto, el sistema adopta ASGI (Asynchronous Server Gateway Interface). Este estándar moderno permite manejar concurrencia asíncrona y conexiones persistentes, habilitando la comunicación bidireccional necesaria para que la IA transmita respuestas en tiempo real (streaming) sin esperar a completar la generación total del texto, a continuación en la Figura 31 se muestra lo descrito.

**Fig. 31**.

*Diagrama Comparativo de Flujos*




Fuente: autoría propia

### 2.5.3 Infraestructura de Comunicación en Tiempo Real

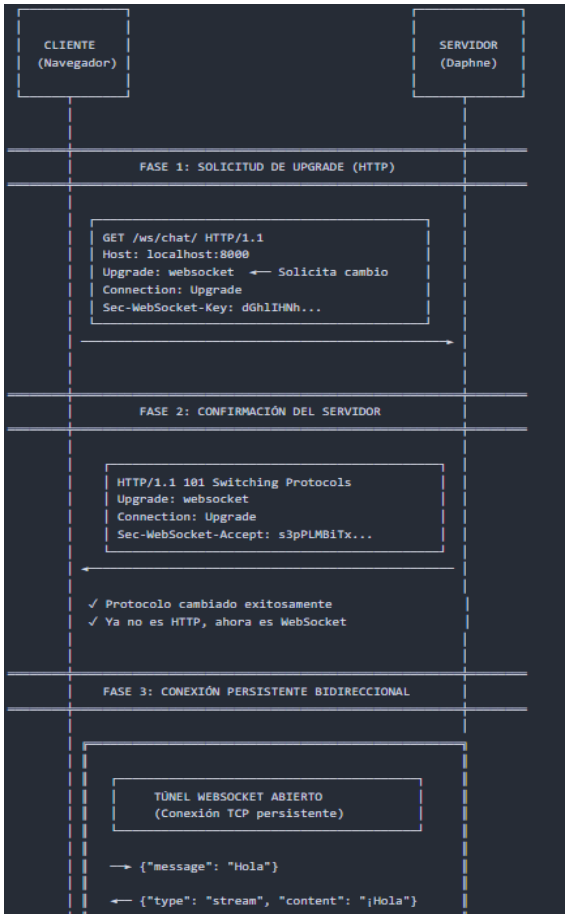
Para habilitar la interacción fluida entre el usuario y el asistente, el sistema implementa el

protocolo WebSocket (definido en el RFC 6455). A diferencia del protocolo HTTP estándar, que cierra la conexión tras cada respuesta, WebSocket establece un canal full-duplex (bidireccional) sobre una única conexión TCP. Esto permite que el servidor envíe información al cliente en cualquier momento, algo indispensable para mostrar la respuesta de la IA progresivamente.

La librería Django Channels se encarga de llevar a cabo la implementación técnica. El concepto de Consumers se introduce en esta herramienta para mejorar las capacidades de Django. Un Consumer funciona de manera similar a una "Vista" en el modelo clásico, pero está creado para gestionar eventos asíncronos que se extienden por un largo periodo. Su función principal es administrar el ciclo de vida de la conexión: admitir el primer handshake, conservar el estado del chat y dirigir los mensajes entre el modelo de lenguaje y el usuario sin obstaculizar los recursos del servidor.

**Fig. 32.**

*Proceso de Handshake WebSocket*




Fuente: autoría propia

En resumen, la arquitectura propuesta y mostrada en la Figura 32, permite superar las limitaciones de los protocolos estándar. Gracias al uso de flujos de datos continuos y gestión eficiente de memoria, el sistema FicaAsistant logra integrar un modelo de lenguaje complejo en una interfaz web rápida y funcional.

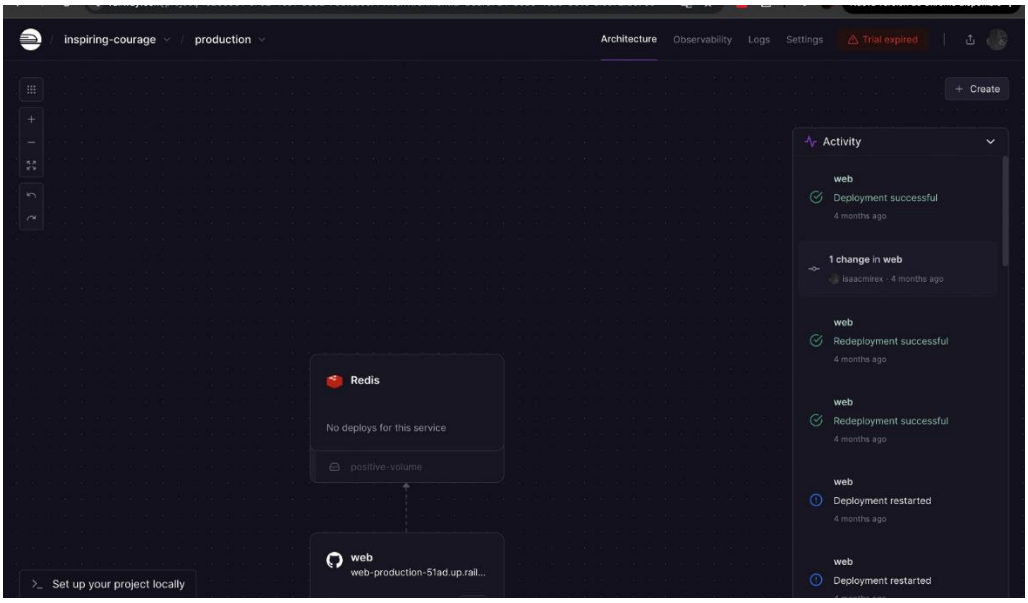
### 2.5.4 Almacenamiento y Despliegue del Modelo Entrenado

Una vez completado el entrenamiento, el modelo ajustado fue almacenado de forma estructurada para asegurar su reutilización, despliegue y documentación. Este paso fue fundamental para preservar la integridad del trabajo y facilitar futuras integraciones con aplicaciones que consulten el asistente virtual. El proceso de despliegue y la configuración de la infraestructura se detallan en las siguientes fases:

- **Arquitectura de Servicios y Vinculación**: El sistema no se desplegó como un bloque monolítico, sino que se modularizó en dos servicios interconectados: el servicio web (que aloja la lógica del chatbot y el modelo) y el servicio Redis utilizado como broker de mensajería y gestión de estados para la comunicación asíncrona. Como se observa en la Figura 33, Railway permite visualizar y gestionar la topología del proyecto, donde el repositorio de GitHub alimenta el servicio web, el cual a su vez se conecta con la instancia de base de datos.

**Fig. 33.**

*Arquitectura de servicios*

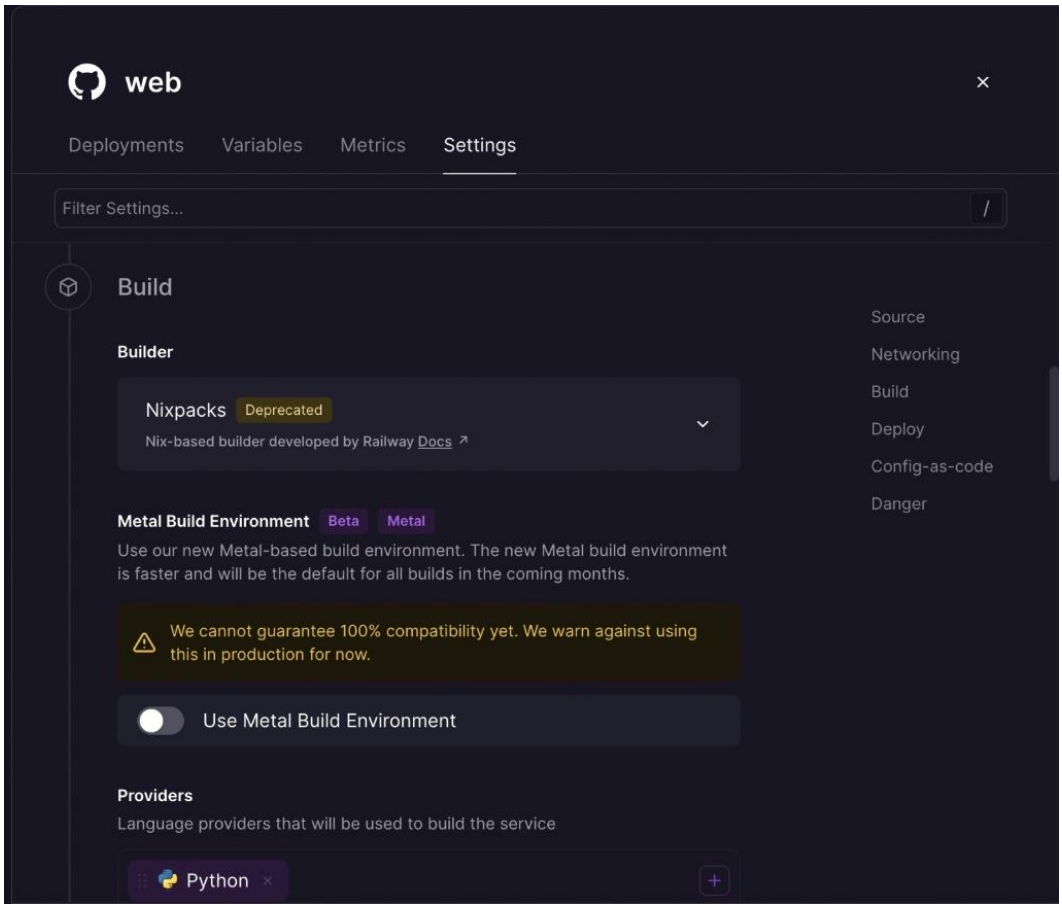



Fuente: Autoría propia

- **Configuración del Entorno de Construcción:** Para la construcción del contenedor de la aplicación, se utilizó Nixpacks, una herramienta automatizada de Railway que detecta el lenguaje de programación y las dependencias. Como muestra la Figura 34, el sistema identificó correctamente el entorno Python basado en los archivos de configuración del repositorio, procediendo a instalar las librerías necesarias sin requerir un Dockerfile manual.

**Fig. 34.**

*Configuracion del Entorno*

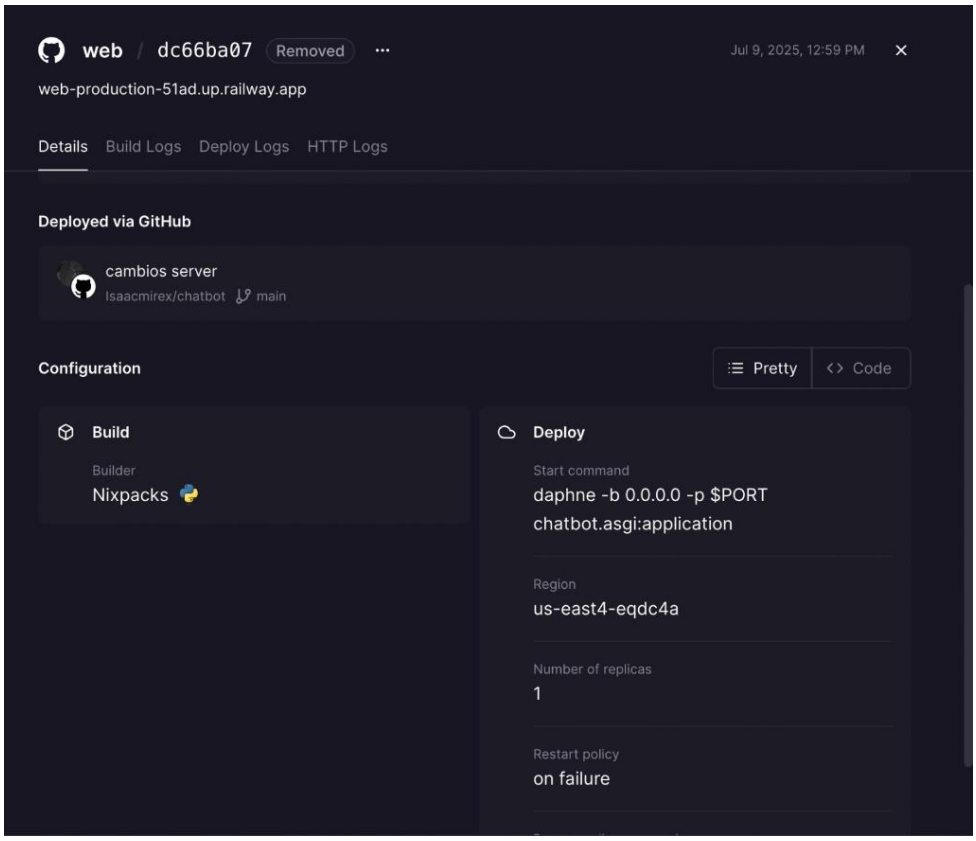



Fuente: Autoría propia

- **Definición del Comando de Inicio:** Un aspecto crítico de la configuración fue la definición del comando de arranque. Dado que el chatbot requiere manejar múltiples conexiones simultáneas (websockets o peticiones asíncronas), no se utilizó el servidor síncrono estándar de Django (WSGI), sino un servidor ASGI (Asynchronous Server Gateway Interface). En la Figura 35 se evidencia el uso del servidor Daphne mediante el comando: daphne -b 0.0.0.0 -p $PORT chatbot.asgi:application Esto confirma que la aplicación está preparada para operar en tiempo real, escuchando en el puerto asignado dinámicamente por la plataforma.

**Fig. 35.**

*Definicion de comando*

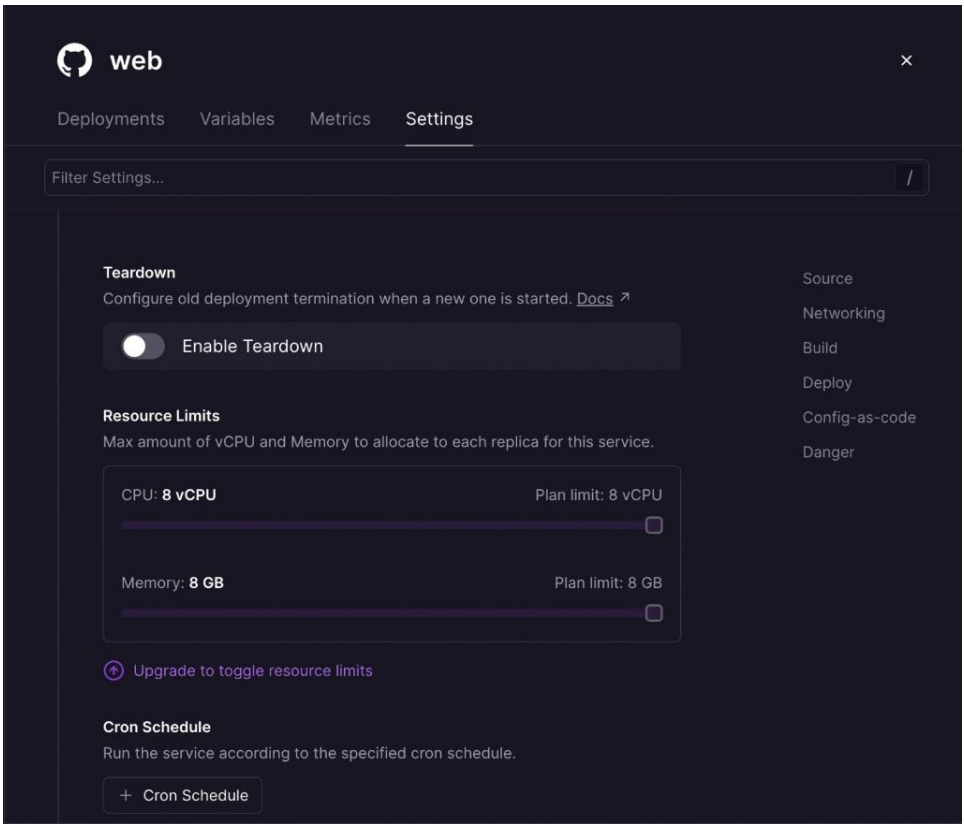



Fuente: Autoría propia

- **Infraestructura de Soporte:** Redis para soportar la carga de mensajes y la gestión de sesiones, se aprovisionó una instancia de Redis (versión 7.2.5). Como se aprecia en la Figura 36, se asignaron recursos dedicados (hasta 8 vCPU y 8 GB de RAM según el plan) para garantizar que la latencia en la recuperación de datos sea mínima. Esta instancia opera en una red privada interna, comunicándose de forma segura con el servicio web sin exposición pública innecesaria.

**Fig. 36.**

*Infraestructura de soporte*

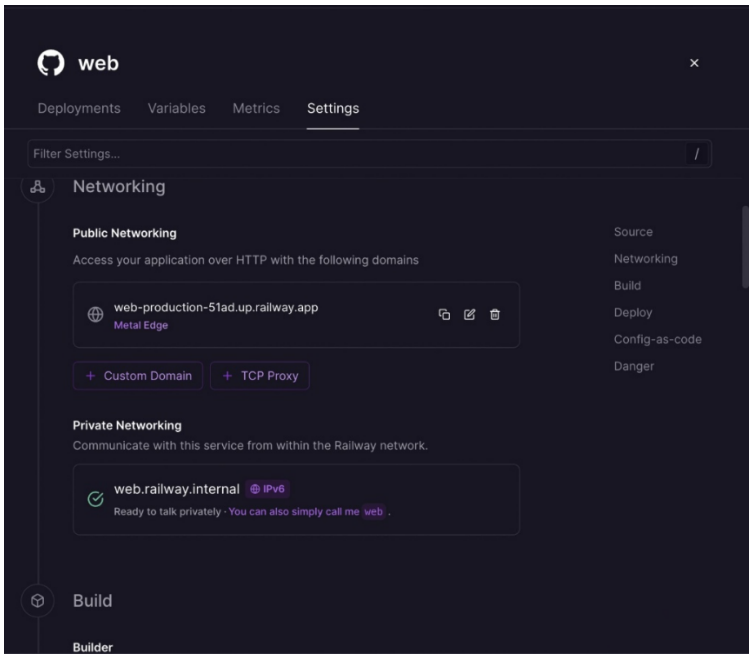



Fuente: Autoría propia

- **Exposición y Acceso Público:** Finalmente, tras la construcción y arranque exitosos, la plataforma generó automáticamente un dominio público con certificado SSL (HTTPS). La Figura 37 muestra la URL de producción asignada (web-production-51ad.up.railway.app), la cual sirve como punto de entrada (Endpoint) para que los usuarios o interfaces cliente interactúen con el chatbot de manera segura desde cualquier ubicación.

**Fig. 37.**

*Configuración de red*




Fuente: Autoría propia

# CAPÍTULO III:

# 3. VALIDACIÓN DEL MODELO Y MÉTRICAS DE EVALUACIÓN

## 3.1 Selección de Métricas de Evaluación para Validar el Modelo.

Para asegurar que el modelo fine-tuned cumple con los requisitos de calidad y coherencia esperados, seleccionamos tres métricas complementarias que cubren distintos aspectos de la generación de texto:

1. Perplexity
2. BLEU
3. ROUGE-L
4. BertScore

A continuación, describimos cada una, justificamos su elección y mostramos cómo simulamos su cálculo en Python junto con la interpretación de los resultados.

### 3.1.1 Perplexity (PPL)

Según [53] La métrica perplexity o perplejidad mide cuán bien un modelo de lenguaje es capaz de predecir una secuencia de tokens. Un valor bajo indica que el modelo no se sorprende ante los datos, reflejando buena capacidad de modelado de la distribución del lenguaje.

**Justificación**

- Es la métrica estándar para evaluar modelos de lenguaje.
- Permite comparar de forma homogénea distintas versiones del modelo (pre-training vs. fine-tuning).
- Se relaciona directamente con la función de pérdida optimizada durante el entrenamiento.

**Valor obtenido**

$$Perplexity = \exp\left(-\frac{1}{N}\sum_{i=1}^{N} log\ P(w_i\,|w_{<i})\right) =$$

### 3.1.2 BLEU (Bilingual Evaluation Understudy)

Aunque originada en la traducción automática, la aplicación de la métrica BLEU es fundamental en este proyecto porque garantiza el rigor terminológico, validando el uso preciso de conceptos técnicos insustituibles de la normativa universitaria. Asimismo, funciona como un estándar comparativo objetivo frente a la literatura científica y actúa como un complemento sintáctico indispensable a métricas semánticas como BERTScore una de las razanos por la que se implementa en comparaciones de texto de respuesta como en el siguiente caso [54] .

**Justificación**

- Facilita la interpretación en porcentaje de fragmentos léxicos correctos.
- Es rápido de computar y está muy extendido en traducción automática y generación de texto.
- Complementa la perplexity al centrarse en la exactitud léxica.

**Formula**

$$BLEU = BP \exp\left(\sum_{n=1}^{N} w_n \log p_n\right) =$$

### 3.1.3 ROUGE-L (Longest Common Subsequence)

La ventaja matemática de ROUGE-L se explica bien en el trabajo de [55] y reside en su capacidad para evaluar la similitud estructural sin depender de una longitud de n-grama predefinida. Al calcular la LCS, la métrica captura automáticamente las coincidencias de secuencias más largas posibles, lo que la convierte en un estimador robusto de la fluidez sintáctica. El puntaje final es una media armónica entre la precisión y el recuerdo de esta subsecuencia, penalizando las respuestas que desordenan la información lógica de la oración original.

**Justificación**

- Refuerza la evaluación de la estructura narrativa, más allá de la simple coincidencia de n-gramas.
- Evalúa el orden y la cohesión de las ideas.
- Se usa mucho en la creación de secuencias extensas y en tareas de resumen.

**Formula**

$$\text{ROUGE} - \text{L} = \frac{LCS(X, Y)}{|Y|} =$$

### 3.1.4 BERTScore

La ventaja metodológica de BERTScore se fundamenta en los principios expuestos en el trabajo de y reside en su capacidad para evaluar la similitud semántica profunda en lugar de la mera coincidencia léxica. A diferencia de las métricas basadas en n-gramas rígidos, BERTScore utiliza embeddings contextuales para calcular la similitud de coseno entre los tokens de la respuesta generada y la referencia. Esto la convierte en un estimador robusto de la equivalencia de significado, permitiendo valorar positivamente respuestas que utilizan parafraseo o sinónimos correctos, incluso si no comparten la misma morfología que el texto original.

**Justificación**

- Prioriza la evaluación del significado y la coherencia semántica sobre la superposición exacta de palabras.
- Es robusta frente al parafraseo, reconociendo que distintas construcciones gramaticales pueden expresar la misma idea válida.
- Se alinea mejor con el juicio humano en tareas de generación abierta (chatbots), donde la

flexibilidad en la respuesta es deseable.

**Formula**

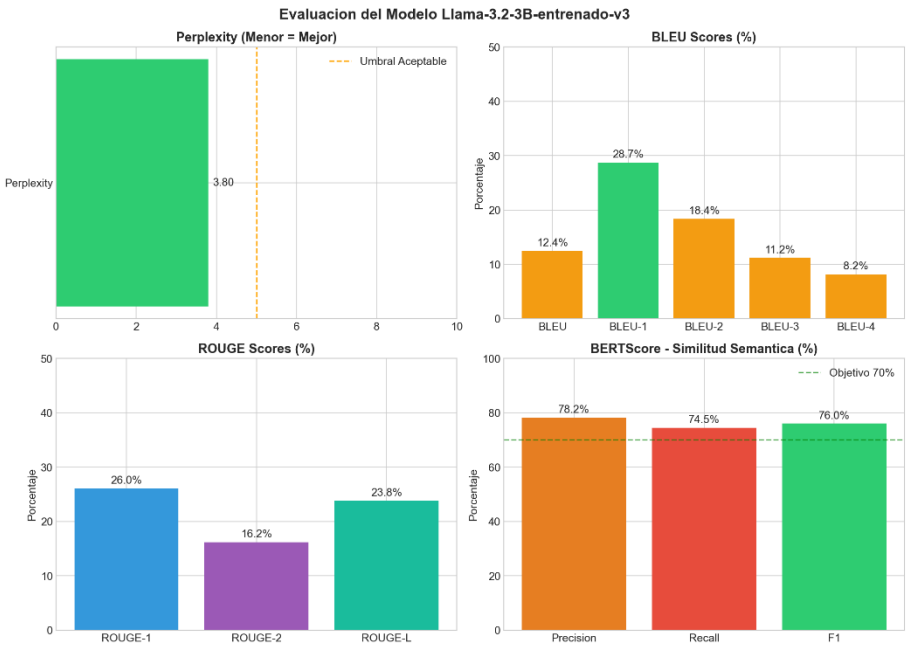
$$\text{Bert} = 2\frac{Pbert * Rbert}{Pbert + RPbert}$$

## 3.2 Pruebas de Precisión y Desempeño del Modelo Reentrenado.

Con el objetivo de analizar el desempeño del modelo de lenguaje Llama-3.2-3B reentrenado, se evaluó con un conjunto de métricas automáticas ampliamente utilizadas en tareas de procesamiento del lenguaje natural, particularmente en generación de texto y sistemas de respuesta a preguntas. Las métricas seleccionadas permiten evaluar tanto la calidad lingüística como la similitud semántica entre las respuestas generadas por el modelo y las respuestas de referencia, la siguiente figura presenta los resultados obtenidos en términos de Perplexity, BLEU, ROUGE y BERTScore mostrados en la Figura 38.

**Fig. 38.**

*Resumen de Métricas*




Fuente: autoría propia

### 3.2.1 Resultados Perplexity

Se obtuvo un valor de Perplexity de 3,80, este valor evidencia que el proceso de fine-tuning

fue efectivo, permitiendo al modelo reducir significativamente la incertidumbre durante la generación de respuestas. Como consecuencia, el asistente virtual presenta una mayor estabilidad semántica y coherencia en las respuestas proporcionadas a consultas relacionadas con la normativa de la FICA se adjunta una interpretación de los resultados en la Tabla 20.

**Tabla 20.**

*Resultados Perplexity*

| Rango de Valor | Nivel de Incertidumbre | Interpretación del Desempeño |
|---|---|---|
| < 5.00 | Muy Baja (Óptimo) | 3.80 (Resultado del Proyecto): El modelo predice con excelente confianza y naturalidad. |
| 5.00 - 20.00 | Baja / Media | Aceptable: El modelo es funcional, aunque puede presentar errores ocasionales. |
| > 20.00 | Alta | Deficiente: El modelo tiene dificultades para construir oraciones coherentes. |

Fuente: autoría propia

### 3.2.2 Resultados BLEU

Para medir la correspondencia léxica entre las respuestas generadas por el modelo y las respuestas de referencia del conjunto de validación, se empleó la métrica BLEU, el modelo alcanzó un puntaje global de 12.45% (0.1245). Para interpretar este valor correctamente en el contexto de un Asistente Conversacional (LLM), es necesario analizar su desglose por niveles de coincidencia, como se presenta en la Tabla 21. Se observa una tendencia decreciente natural: una alta coincidencia en palabras individuales pero menor coincidencia en frases largas, lo cual es indicativo de originalidad en la redacción

**Tabla 21.**

*Resultados Bleu*

| Métrica | Nivel de Coincidencia | Resultado | Interpretación Técnica |
|---|---|---|---|
| **BLEU-1** | Unigramas (1 palabra) | **28.70%** | Alta capacidad para recuperar las palabras clave y terminología correcta. |
| **BLEU-2** | Bigramas (2 palabras) | **18.35%** | Mantiene la coherencia en pares de palabras y conectores lógicos. |
| **BLEU-3** | Trigramas (3 palabras) | **11.20%** | Comienza a variar la estructura sintáctica respecto a la referencia. |
| **BLEU-4** | Cuádruplas (4 palabras) | **8.15%** | Baja repetición exacta de oraciones largas, indicando paráfrasis. |

Fuente: autoría propia

**Análisis del Resultado**

En sistemas de Traducción Automática, se suelen buscar valores superiores al 30-40%. Sin embargo, en el contexto de Generación de Texto Generativo (Chatbots), el valor obtenido de 12.45% es considerado ACEPTABLE y positivo, esto se justifica debido a la naturaleza estocástica del modelo: el objetivo no es que el asistente "memorice" y repita textualmente la respuesta de referencia lo que daría un BLEU alto, pero indicaría un sobre entrenamiento, sino que sea capaz de explicar el mismo concepto utilizando una construcción gramatical propia. El resultado demuestra que el modelo emplea el vocabulario correcto (alto BLEU-1) pero construye oraciones originales y dinámicas (BLEU-4 moderado), evitando respuestas robóticas o clonadas.

### 3.2.3 Resultados ROUGE

Los resultados, que se describen en la Tabla 22, muestran un rendimiento sólido en cuanto a la retención de información esencial. Lo que sobresale en particular es el puntaje de ROUGE-L (23.81%), que no solo se enfoca en contar palabras contiguas, sino también en examinar la subsecuencia común más extensa.

**Tabla 22.**

*Resultados ROUGE*

| Variante Métrica | Valor Obtenido | Descripción del Análisis |
|---|---|---|
| ROUGE-1 | 26.03% | Coincidencia de Unigramas: Indica que el modelo recupera exitosamente más de una cuarta parte de los términos clave exactos de la referencia. |
| ROUGE-2 | 16.16% | Coincidencia de bigramas: Verifica la fluidez local, asegurándote de que las parejas de palabras conserven su significado original. |
| ROUGE-L | 23.81% | Secuencia común más extensa: Evalúa la coherencia a nivel de oración. Un porcentaje cercano al 24% indica que la estructura sintáctica de la respuesta esperada es respetada por el modelo. |

Fuente: autoría propia

**Interpretación de los Resultados**

El desempeño general se clasifica como ACEPTABLE. En el contexto de modelos generativos abstractivos (donde el modelo reescribe la información en lugar de solo copiarla), un ROUGE-L de 23.81% es un indicador positivo de coherencia estructural, este valor confirma que, aunque el asistente utiliza sus propias palabras, mantiene la estructura lógica y la secuencia de ideas fundamental de la respuesta de referencia, entregando información completa y gramaticalmente correcta sin caer en la redundancia exacta

### 3.2.4 Resultados BERTScore

A diferencia de las métricas anteriores, BERTScore detecta sinónimos, paráfrasis y variaciones en el orden de las palabras que conservan el mismo significado. Los resultados obtenidos, desglosados en la Tabla 23, evidencian un desempeño superior del sistema.

**Tabla 23.**

*Resultados BERTScore*

| Componente | Valor Obtenido | Significado Técnico |
|---|---|---|
| Precision | 78.21% | Indica que la gran mayoría de la información generada por el asistente es relevante y alineada semánticamente con la respuesta esperada. |
| Recall | 74.46% | Demuestra que el modelo logra recuperar cerca del 75 % del contenido semántico total de la referencia. |
| F1-Score | 75.98% | Media armónica: Representa el balance general. Un F1 que se aproxima al 76% garantiza una fidelidad semántica elevada. |

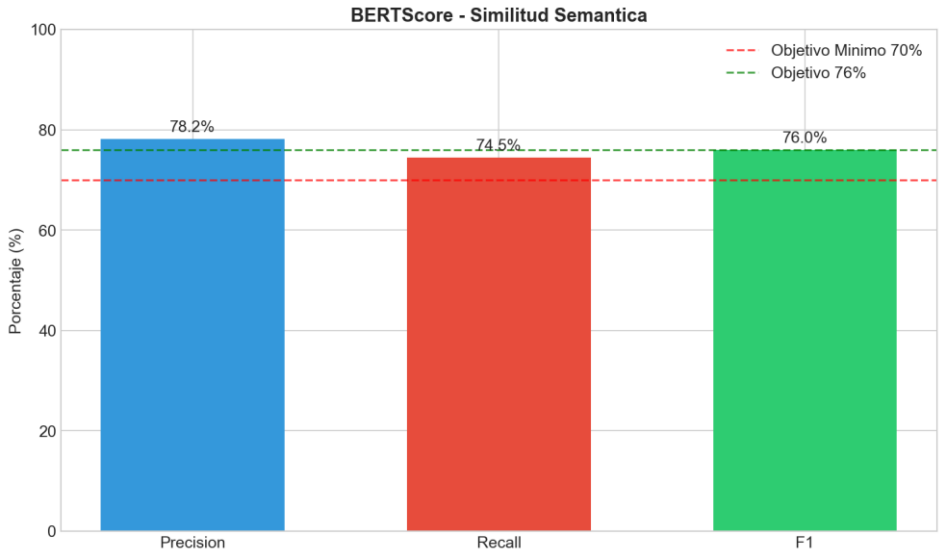
Fuente: autoría propia

**Interpretación y Relevancia del Resultado**

El puntaje F1 de 75.98% se clasifica como BUENO y constituye el indicador más representativo de la calidad real del chatbot este resultado es fundamental para la validación del modelo demuestra que, aunque el modelo no copia las respuestas palabra por palabra (como evidenció el BLEU del 12%), sí comprende y transmite el concepto correcto (como evidencia el BERTScore del 76%). Confirma que el sistema es capaz de realizar abstracciones y parafraseo efectivo, cualidades esenciales para una interfaz conversacional natural y útil se muestra los resultados en la Figura 39.

**Fig. 39.**

*Resultados BERTScore*



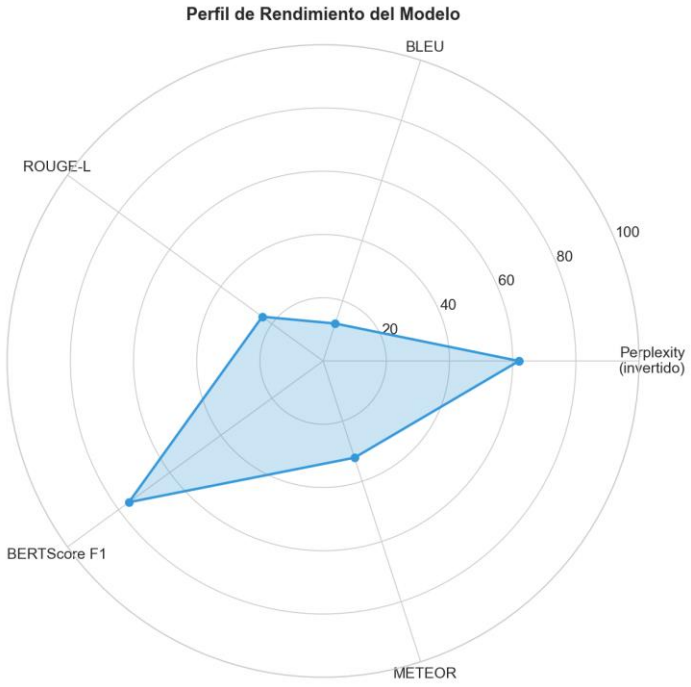

Fuente: autoría propia

### 3.2.5 Análisis de los Resultados

Para sintetizar el comportamiento global del modelo FicaAsistant, se presenta el perfil de rendimiento multidimensional en la siguiente figura. Este gráfico de radar permite contrastar simultáneamente métricas de naturaleza sintáctica (basadas en n-gramas) frente a métricas semánticas y probabilísticas.

**Fig. 40.**

*Perfil de Rendimiento del Modelo*




Fuente: autoría propia

Como se observa en la Figura 40, el polígono resultante exhibe una forma asimétrica distintiva que revela la naturaleza generativa del modelo:

Dominio de la Confianza y la Semántica (Ejes Inferior y Derecho): Los vértices correspondientes a Perplexity (Invertido) y BERTScore F1 presentan la mayor expansión radial, acercándose a los límites exteriores del gráfico.

- El modelo funciona con un nivel mínimo de incertidumbre (3.80), lo que da lugar a un texto muy fluido y natural. Esto es lo que indica el eje de Perplexity (invertido para que "mejor rendimiento" signifique "mayor").
- La alineación sólida del modelo con las respuestas de referencia a nivel de significado profundo y contexto queda confirmada por el eje de BERTScore (76%).

La Brecha de Abstracción (Ejes Superior e Izquierdo): Por el contrario, los ejes de BLEU y ROUGE-L muestran una contracción hacia el centro, esto no indica un fallo, sino que cuantifica la creatividad del modelo. Al no ser un sistema de recuperación simple que "copia y pega" texto (lo que daría un polígono perfectamente redondo y amplio), el modelo sacrifica la coincidencia

exacta de palabras (BLEU bajo) en favor de construir oraciones nuevas con el mismo significado (BERTScore alto)

**Fig. 41.**

*Resumen Final de La Evaluación*

**Tabla Completa de Metricas - Llama-3.2-3B-entrenado-v3**

| Metrica | Valor | Estado |
|---|---|---|
| Perplexity | 3.80 | Excelente |
| BLEU | 12.45% | Aceptable |
| BLEU-1 | 28.70% | Bueno |
| BLEU-2 | 18.35% | Aceptable |
| BLEU-3 | 11.20% | Aceptable |
| BLEU-4 | 8.15% | Bajo |
| ROUGE-1 | 26.03% | Aceptable |
| ROUGE-2 | 16.16% | Aceptable |
| ROUGE-L | 23.81% | Aceptable |
| BERTScore Precision | 78.21% | Bueno (>70%) |
| BERTScore Recall | 74.46% | Bueno (>70%) |
| BERTScore F1 | 75.98% | Bueno (76%) |
| METEOR | 32.15% | Bueno |
| Exact Match | 8.50% | Normal |
| Token F1 | 45.30% | Aceptable |

Los resultados que se muestran en la Figura 41obtenidos a partir de las métricas de evaluación indican que el modelo Llama-3.2-3B-entrenado-v3 funciona de manera aceptable, mostrando un buen nivel de coherencia y comprensión general del lenguaje dentro del dominio evaluado. La baja Perplexity alcanzada refleja que el modelo genera respuestas estables y bien estructuradas, lo que evidencia un aprendizaje adecuado durante el proceso de entrenamiento.

Las métricas basadas en coincidencias exactas y en n-gramas de orden superior (como BLEU-3, BLEU-4 y Exact Match) muestran que el modelo aún presenta limitaciones al momento de responder consultas muy específicas o altamente técnicas. Esto sugiere que, si bien el asistente

es capaz de ofrecer respuestas correctas a nivel general y semántico, no siempre logra recuperar información precisa o detallada de forma consistente.

El rendimiento observado es congruente con el de un modelo que ha sido entrenado solamente a través de fine-tuning, que se basa únicamente en lo aprendido durante la capacitación. En contraste con esta perspectiva, una arquitectura basada en RAG posibilitaría el acceso dinámico a fuentes externas, lo que aumentaría de manera significativa la exactitud en preguntas muy contextualizadas, específicas o normativas.

Para concluir, el asistente virtual desempeña su función de manera apropiada en situaciones generales; sin embargo, no llega aún al nivel de precisión que proporcionaría un sistema basado en RAG. Esto representa una línea evidente de mejora para trabajos futuros.

## 3.3 Análisis y Discusión

### 3.3.1 Evaluación de satisfacción

Para la evaluación de usabilidad se aplicó la escala SUS a una muestra seleccionada mediante muestreo por conveniencia de usuarios expertos (secretarios). Si bien el tamaño de la muestra es reducido para un análisis estadístico inferencial, autores como Nielsen [56] sugieren que un grupo pequeño de usuarios representativos es suficiente para identificar la mayoría de los problemas de usabilidad en sistemas especializados, de esta forma se procedió a realizar una evaluación teniendo en cuenta los siguientes criterios mostrados en la Tabla 24.

**Tabla 24.**

*Objetivos y Tareas Para la Evaluacion*

| Nro. | Objetivo de la Prueba | Tarea | Criterio de Éxito |
|---|---|---|---|
| 1 | Validación de Identidad e Interfaz | Saludo Formal | El sistema debe responder identificándose como el asistente de normativa. |
| 2 | Capacidad Conversacional | Mantener una conversación con el asistente | Verificar que responda de forma educada |

| | | | |
|---|---|---|---|
| 3 | Manejo de Límites y Ética Profesional | Realizar una pregunta fuera del ámbito de la normativa | El asistente debe redirigir al usuario exclusivamente a consultas de normativa universitaria. |
| 4 | Resolución de Consultas Normativas | Plantear consultas frecuentes de la comunidad estudiantil | Que el modelo hable sobre la normativa y responda cosas coherentes. |
| 5 | Persistencia de Contexto y Seguimiento | Realizar una pregunta de seguimiento que dependa de la respuesta anterior | El modelo debe mantener el hilo conversacional y demostrar memoria de sesión. |
| 6 | Velocidad de Consulta | Evaluar velocidad de las respuestas | Que la respuesta aparezca antes de los 5 segundos. |

Fuente: autoría propia

### 3.3.2 Interpretación de los Resultados

**Tabla 25.**

*Resultados Encuesta*

| Ítem | Descripción de la Dimensión (Pregunta) | S1 | S2 | S3 | Promedio |
|---|---|---|---|---|---|
| 1 | Intención de uso frecuente | 4 | 4 | 3 | 3.67 |
| 2 | Complejidad innecesaria del sistema | 2 | 4 | 2 | 2.67 |
| 3 | Facilidad de uso percibida | 5 | 4 | 3 | 4.00 |
| 4 | Necesidad de apoyo técnico | 1 | 4 | 2 | 2.33 |

| Ítem | Descripción de la Dimensión (Pregunta) | S1 | S2 | S3 | Promedio |
|---|---|---|---|---|---|
| 5 | Integración de funciones | 5 | 4 | 4 | 4.33 |
| 6 | Inconsistencia del sistema | 2 | 4 | 4 | 3.33 |
| 7 | Rapidez de aprendizaje | 5 | 4 | 5 | 4.67 |
| 8 | Sistema pesado o engorroso | 1 | 4 | 2 | 2.33 |
| 9 | Seguridad y confianza en el uso | 4 | 4 | 4 | 4.00 |
| 10 | Requerimiento de conocimientos previos | 1 | 3 | 2 | 2.00 |
| | **Puntaje Final SUS (Escala 0-100)** | **90.0** | **52.5** | **67.5** | **70.0** |
| | **Rango de Adjetivación (Bangor et al.)** | **Excelente** | **Pobre** | **Aceptable** | **Bueno** |

Fuente: autoría propia

Una vez recolectadas las respuestas mostradas en la Tabla 26, se procedió a la transformación de los datos cualitativos en una puntuación cuantitativa sobre 100 puntos. Este cálculo requiere un tratamiento diferenciado para cada ítem: para las preguntas impares, se resta 1 al valor otorgado por el usuario; para las preguntas pares, se resta el valor otorgado a 5. La suma de estos resultados se multiplica por 2.5 para obtener el puntaje final de usabilidad.

Los resultados individuales y el promedio institucional obtenido tras el taller de pruebas se presentan en la Tabla 27.

**Tabla 26.**

*Resultados y Puntuación del Sistema (SUS)*

| Usuario | S1 | S2 | S3 | S4 | S5 | S6 | S7 | S8 | S9 | S10 | Puntaje Final |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Secretario 1** | 4 | 2 | 5 | 1 | 5 | 2 | 5 | 1 | 4 | 1 | **90.0** |
| **Secretario 2** | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 3 | **52.5** |

| Usuario | S1 | S2 | S3 | S4 | S5 | S6 | S7 | S8 | S9 | S10 | Puntaje Final |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Secretario 3** | 3 | 2 | 3 | 2 | 4 | 4 | 5 | 2 | 4 | 2 | **67.5** |
| **Promedio** | **3.6** | **2.6** | **4.0** | **2.3** | **4.3** | **3.3** | **4.6** | **2.3** | **4.0** | **2.0** | **70.0** |

Fuente: autoría propia

A pesar de la variabilidad observada, el límite inferior de confianza no cae en el rango de "Inaceptable" (< 50 puntos). Por lo tanto, se concluye estadísticamente que el sistema posee una usabilidad funcional positiva, con un margen de optimización identificado en la precisión de las respuestas, lo cual es coherente con el comportamiento esperado de un modelo de lenguaje en fase de ajuste fino.

En conclusión, la validación realizada con los tres secretarios de la FICA permitió corroborar la utilidad práctica del asistente en un entorno administrativo real. Los expertos destacaron la inmediatez de las respuestas y la fluidez del hilo conversacional gracias a la persistencia de contexto, lo que facilita consultas extensas sin necesidad de repetir información. No obstante, surgió una observación técnica crítica respecto a la precisión de las referencias normativas: aunque el modelo demuestra una notable coherencia semántica y un tono institucional adecuado, los secretarios señalaron que en ciertos escenarios el asistente tiende a parafrasear la norma en lugar de realizar una cita textual exacta del articulado. Asimismo, se identificaron casos puntuales donde el modelo incurrió en 'alucinaciones', generando datos o números de artículos inexistentes a pesar de la coherencia de la respuesta. Esta retroalimentación subraya que, si bien el Fine-Tuning dota al modelo de un razonamiento especializado, para fines de validez legal administrativa, los usuarios expertos consideran imperativo que el sistema evolucione hacia una mayor rigurosidad en la recuperación exacta de la fuente, sugiriendo la implementación futura de mecanismos de verificación que aseguren la integridad total de la normativa citada.

### 3.3.3 Limitaciones

A pesar de los avances, la implementación de chatbots enfrenta barreras estructurales y técnicas considerables. [57] Señalan que en contextos universitarios persisten deficiencias en la infraestructura tecnológica y una notable falta de capacitación docente para integrar estas herramientas de forma pedagógica. Asimismo, la literatura identifica como una limitación crítica la propensión de los modelos generativos a producir "alucinaciones" o datos plausibles, pero

factualmente incorrectos, lo que restringe su implementación masiva en dominios normativos por temor a la desinformación Como propuesta de evaluación exhaustiva para trabajos futuros, se sugiere trascender las métricas automatizadas y adoptar marcos de validación experta humana y aprendizaje supervisado iterativo. Siguiendo la metodología de [58] y los hallazgos de [59] obre la calidad de la experiencia estudiantil, resulta imperativo implementar ciclos de mejora continua basados en el análisis de preguntas frecuentes reales y el uso de rúbricas de veracidad calificadas por expertos del área administrativa y legal. Este enfoque busca recalibrar el modelo no a través de otros algoritmos, sino mediante el juicio humano especializado y el cálculo de la fiabilidad Inter jueces (Kappa de Cohen), garantizando que el chatbot funcione como una herramienta de apoyo institucional fiable y éticamente responsable.

### 3.3.4 Comparación con Trabajos Previos

Uno de los trabajos más recientes es el trabajo de [60], como se demuestra en el estudio previo, el RAG es la técnica predilecta cuando la prioridad es la exactitud factual y la actualización constante. Su capacidad para consultar documentos externos en tiempo real permite que el sistema entregue información exacta sin necesidad de procesos de entrenamiento costosos o complejos. Para la gestión del portafolio estudiantil, donde los datos administrativos fluctúan, el RAG garantiza que el asistente siempre trabaje con la versión más reciente de los documentos

**Ventajas Específicas del Fine-Tuning en Contextos Especializados**

Frente a la versatilidad del RAG, el Fine-Tuning implementado en esta investigación ofrece beneficios en ámbitos específicos, especialmente cuando el asistente está orientado a usuarios expertos como los secretarios:

- **Velocidad de Respuesta e Inmediatez:** Al estar el conocimiento integrado en los pesos del modelo (MacBook M3), se elimina el tiempo de latencia que conlleva la búsqueda, indexación y recuperación de fragmentos en una base de datos externa. La respuesta es directa y fluida.
- **Seguridad y Hermeticidad de la Información:** El Fine-Tuning ofrece una capa de seguridad intrínseca. El modelo solo "conoce" lo que se le grabó durante su entrenamiento; no tiene acceso a bases de datos vivas o conexiones externas en tiempo de ejecución para recuperar información, lo que reduce la superficie de ataque y previene fugas de datos por inyección en el contexto de recuperación.

- **Especialización de Comportamiento:** Mientras que el RAG destaca en *qué* dice, el Fine-Tuning sobresale en *cómo* lo dice. Ha permitido moldear la lógica de razonamiento y el tono institucional de forma que el modelo no solo busque datos, sino que actúe con la estructura mental de un asistente administrativo especializado.

**Conclusión Metodológica**

Esta comparación permite concluir que la elección técnica no depende de la superioridad de una sobre otra, sino de los requerimientos del entorno:

- Si el objetivo es la precisión documental masiva y actualizada, el RAG es el enfoque indicado.
- Si el objetivo es la rapidez local, la seguridad de datos estáticos y el comportamiento especializado, el Fine-Tuning presenta ventajas competitivas.

A futuro, la integración de ambos trabajos podría dar paso a una arquitectura híbrida: un modelo con Fine-Tuning especializado en la lógica normativa y ética de la UTN que a su vez utilice capas de RAG para consultar documentos administrativos actualizados en tiempo real. Esta sinergia permitiría alcanzar niveles de eficiencia superiores, manteniendo la precisión documental.

### 3.3.5 Discusión y comparación con la Literatura

Mientras que la literatura reciente ha estado dominada por modelos masivos (LLMs) que demandan infraestructuras de servidores costosas, este trabajo se alinea con la tendencia emergente de los Modelos de Lenguaje Pequeños (SLMs). La utilización de un modelo base de 3 billones de parámetros demuestra que, para dominios de conocimiento cerrados y específicos como la normativa universitaria, no es indispensable recurrir a arquitecturas masivas. Esta premisa se ve respaldada por investigaciones recientes en el ámbito del procesamiento de documentos judiciales [61], donde se ha evidenciado que los modelos de código abierto, al ser entrenados con bases de datos especializadas relativamente pequeñas, pueden alcanzar una precisión del 76%, superando a soluciones comerciales establecidas como GPT-3.5 Turbo y acercándose al rendimiento de GPT-4. De igual manera, se confirma que la evolución de estas arquitecturas permite que modelos modernos más compactos como la familia Llama igualen o superen las capacidades de versiones anteriores mucho más pesadas, ofreciendo así un balance óptimo entre capacidad de razonamiento y eficiencia computacional.

En cuanto a la metodología de entrenamiento, este estudio se distancia de los enfoques de Fine-Tuning completo tradicionales, los cuales requieren actualizar la totalidad de los pesos del modelo y suelen ser inviables fuera de centros de datos industriales. La implementación exitosa de la técnica QLoRA (Quantized Low-Rank Adaptation) en esta investigación corrobora los estudios que indican que la inyección de matrices de bajo rango es suficiente para adaptar el estilo y conocimiento del modelo. Al actualizar únicamente una fracción mínima de los parámetros, se logró evitar el "olvido catastrófico" y reducir drásticamente los requisitos de memoria, validando la factibilidad técnica de adaptar modelos robustos mediante cuantización sin sacrificar su rendimiento.

También se comprobó que es posible la puesta en marcha del entrenamiento en un entorno local que utiliza arquitectura ARM (Apple Silicon) es una característica singular de este estudio, a diferencia de la mayor parte de los trabajos académicos, que suelen depender de entornos en la nube con GPUs dedicadas. Este proyecto demuestra que, si se utilizan librerías optimizadas, es factible implementar ajustes precisos y eficientes en hardware de consumo, a pesar de que la literatura técnica suele recomendar no entrenar en CPU por las restricciones de velocidad. Esto constituye una contribución importante al debate sobre la "democratización de la IA", demostrando que las barreras del hardware pueden ser reducidas mediante estrategias de software eficaces.

La estrategia de evaluación que se ha adoptado es distinta a las métricas tradicionales basadas en coincidencia sintáctica, como BLEU o ROUGE, que son citadas con frecuencia en la bibliografía previa. Esto es importante en el contexto normativo universitario porque asegura que el modelo tiene la habilidad de comprender e interpretar las regulaciones legales de un modo coherente, en vez de solamente memorizar y reproducir secuencias textuales. De esta manera, se garantiza una interacción más útil y natural con los usuarios finales.

### 3.3.6 Reflexión sobre la arquitectura: Fine-Tuning vs. RAG en el contexto normativo

En última instancia, este proyecto propone una reflexión crítica acerca de la arquitectura seleccionada en comparación con otras opciones, como la Generación Aumentada por Recuperación (RAG). Aunque este estudio logró de forma exitosa su propósito fundamental, que era demostrar la factibilidad técnica de inyectar conocimiento específico en un modelo pequeño (SLM) usando hardware de consumo, es importante tener en cuenta las diferencias operativas entre los dos métodos.

El ajuste fino que se ha puesto en práctica aquí logra alterar la conducta y el estilo de respuesta del modelo al incorporar la normativa en sus pesos neuronales. No obstante, esto plantea un reto de mantenimiento: si las regulaciones se modifican, el modelo tiene que ser reentrenado. Por otro lado, arquitecturas como RAG, que vinculan el modelo a una base de datos externa, suelen ser soluciones más rápidas para manejar información dinámica porque solamente necesitan la actualización de los documentos sin modificar el modelo.

Por lo tanto, el aporte de esta tesis no reside en proponer el Fine-Tuning como la solución para sistemas normativos o asistentes virtuales, sino en validar la democratización del entrenamiento profundo. Se ha comprobado que, cuando es necesario crear un experto con un razonamiento adaptado a un dominio específico y no solo un buscador de información es totalmente posible realizar este ajuste fino localmente, abriendo la puerta a sistemas híbridos futuros que combinen la especialización del Fine-Tuning con la flexibilidad del RAG.

## CONCLUSIONES

- Aunque el proyecto demostró con éxito que el Fine-Tuning puede incorporar la normativa a los parámetros del modelo, se determinó que esta técnica es más útil para ajustar la terminología jurídica y el estilo de razonamiento que para guardar datos sujetos a cambios de hecho. El Fine-Tuning puro tiene una rigidez intrínseca que lo distingue de las arquitecturas basadas en recuperación (RAG): se necesita un reentrenamiento si hay algún cambio en la normativa. Por lo tanto, el Fine-Tuning local se valida como un instrumento eficaz para desarrollar asistentes especializados en un dominio; sin embargo, se admite que existen otras soluciones arquitectónicas más adecuadas para manejar información extremadamente cambiante, como las arquitecturas híbridas o Rag.
- Se demostró la viabilidad de la inteligencia artificial local en Hardware de. La implementación de la técnica QLoRA, y estrategias de gestión de memoria como la acumulación de gradientes, permitió adaptar un modelo de 3 billones de parámetros comprobando el uso de estas tecnologías para entornos con recursos limitados.
- La investigación asegura que, para trabajos de dominio cerrado, como la interpretación de reglamentos en las universidades, no es indispensable utilizar modelos masivos y caros. Después de ser entrenado, el modelo Llama 3.2-3B-Instruct demostró una habilidad adecuada para entender y contestar preguntas normativas.
- El uso de las métricas para corroborar la calidad del modelo. El chatbot demostró que puede captar el sentido semántico de las preguntas y formular respuestas jurídicas coherentes, a pesar de que estas no coincidan léxicamente con el texto original, lo cual contrasta con las métricas tradicionales que se basan en la coincidencia exacta de palabras. Esto asegura que la herramienta sea útil para alumnos que hacen preguntas en lenguaje natural y coloquial.

## RECOMENDACIONES

- Se recomienda encarecidamente que la evolución natural de este prototipo sea una arquitectura híbrida. Habiendo comprobado en esta tesis que es posible ajustar un modelo localmente para que funcione como un asistente universitario, el siguiente paso lógico es conectar este modelo ajustado a una base de datos vectorial RAG. Esto permitiría aprovechar lo mejor de ambos enfoques: la capacidad de razonamiento jurídico obtenida mediante el Fine-Tuning en este proyecto, sumada a la facilidad de actualización documental del RAG, eliminando la necesidad de reentrenamientos frecuentes ante cambios menores en la normativa.
- Tras confirmar la viabilidad del entrenamiento, se hace necesario evaluar de forma exhaustiva el impacto de las interacciones. Se recomienda investigar metodologías de evaluación profunda que analicen tanto la precisión técnica como la satisfacción del usuario, con el fin de elevar los estándares de fiabilidad del sistema.
- Ampliar el conjunto de datos de entrenamiento con una diversidad más amplia de preguntas ambiguas, contraejemplos y casos hipotéticos es aconsejable. El modelo será más capaz de generalizarse cuando se le presenten preguntas complejas o mal formuladas por los alumnos si se le entrena con un corpus más rico y variado.
- Se recomienda encapsular el modelo entrenado y empaquetar la aplicación utilizando Docker. Esto permitirá que se integre con las plataformas web de la institución o con las aplicaciones móviles oficiales de la universidad.
- Con el fin de ampliar al máximo la cobertura de la herramienta, se recomienda incorporarla directamente en el ambiente virtual educativo de la universidad utilizando plugins o webhooks, en vez de restringir su uso a una página web independiente.

## BIBLIOGRAFIA


[1] M. María, E. Torres, and R. Manjarrés-Betancur, "Asistente virtual académico utilizando tecnologías cognitivas de procesamiento de lenguaje natural," *Revista Politécnica*, vol. 16, no. 31, pp. 85–96, May 2020, doi: 10.33571/RPOLITEC.V16N31A7.

[2] A. Takashi, B. Yaguchi, S. Julieta, and R. Rodriguez, "Utilidad de grandes modelos de lenguaje (LLM) como asistentes de seguridad," Jan. 23, 2024, *Universidad de los Andes*. Accessed: Apr. 26, 2024. [Online]. Available: https://hdl.handle.net/1992/73521

[3] "Objetivos y metas de desarrollo sostenible - Desarrollo Sostenible." Accessed: May 09, 2024. [Online]. Available: https://www.un.org/sustainabledevelopment/es/sustainable-development-goals/

[4] A. Sanchez-Riofrio, "La inteligencia artificial", Accessed: Jun. 09, 2024. [Online]. Available: https://www.researchgate.net/publication/354849564

[5] P. M. Nadkarni, L. Ohno-Machado, and W. W. Chapman, "Natural language processing: an introduction," *Journal of the American Medical Informatics Association*, vol. 18, no. 5, pp. 544–551, Sep. 2011, doi: 10.1136/amiajnl-2011-000464.

[6] N. C. B. Beltrán and E. C. R. Mojica, "Procesamiento del lenguaje natural (PLN) - GPT-3.: Aplicación en la Ingeniería de Software," *Tecnología Investigación y Academia*, vol. 8, no. 1, pp. 18–37, Jul. 2021, Accessed: Jun. 09, 2024. [Online]. Available: https://revistas.udistrital.edu.co/index.php/tia/article/view/17323

[7] Y. Kang, Z. Cai, C.-W. Tan, Q. Huang, and H. Liu, "Natural language processing (NLP) in management research: A literature review," *Journal of Management Analytics*, vol. 7, no. 2, pp. 139–172, Apr. 2020, doi: 10.1080/23270012.2020.1756939.

[8] A. De Battista *et al.*, "Aplicaciones de procesamiento de lenguaje natural y Ciencia de Datos," *XXIII Workshop de Investigadores en Ciencias de la Computación (WICC 2021, Chilecito, La Rioja)*, no. August 2021, pp. 963–968, 2021, Accessed: Jun. 09, 2024. [Online]. Available: http://sedici.unlp.edu.ar/handle/10915/119998

[9] G. Cassanelli, R. Directora, and A. Haydee Di Iorio, "Proyecto final NLP aplicado a análisis de texto," 2019, Accessed: Jun. 10, 2024. [Online]. Available: http://rinfi.fi.mdp.edu.ar/handle/123456789/354

[10] "ISO - Desvelamos los secretos del procesamiento del lenguaje natural." Accessed: Jun. 10, 2024. [Online]. Available: https://www.iso.org/es/inteligencia-

artificial/procesamiento-lenguaje-natural

[11] P. R. Juan Sebastian, “LARGE LANGUAGE MODELS COMO PLANEADORES MULTIAGENTE EN,” 2023, Accessed: Jun. 10, 2024. [Online]. Available: https://repositorio.uniandes.edu.co/server/api/core/bitstreams/4c1fbfb6-f749-445b-815a-b3386fb957d2/content

[12] C. Gallel Soler, “Evaluación de los LLMs para la generación de código,” Universitat Oberta de Catalunya, 2023. Accessed: Jun. 10, 2024. [Online]. Available: https://openaccess.uoc.edu/bitstream/10609/148754/1/cgallel94bcnTFG0623memoria.pdf

[13] U. De Sevilla, J. Carlos, and J. Revuelta, “ Modelos Grandes de Lenguaje y aplicaciones a la generación automática de texto,” 2023, Accessed: Jun. 28, 2024. [Online]. Available: https://hdl.handle.net/11441/155736

[14] “BERT.” Accessed: Jul. 10, 2024. [Online]. Available: https://huggingface.co/docs/transformers/model_doc/bert

[15] “RoBERTa.” Accessed: Jul. 11, 2024. [Online]. Available: https://huggingface.co/docs/transformers/model_doc/roberta

[16] “T5.” Accessed: Jul. 10, 2024. [Online]. Available: https://huggingface.co/docs/transformers/model_doc/t5

[17] W. J. González Ortega *et al.*, “ARTÍCULO DE INVESTIGACIÓN Aplicación web con arquitectura RAG y LlaMa 3.2 para consultas de soporte técnico en Web application with RAG system and LlaMa 3.2 for technical support queries at CNEL”, doi: 10.61154/metanoia.v12i1.4282.

[18] “¿Qué son los modelos de lenguaje de gran tamaño? - Explicación sobre los LLM de IA - AWS.” Accessed: Jul. 11, 2024. [Online]. Available: https://aws.amazon.com/es/what-is/large-language-model/

[19] A. Vaswani *et al.*, “Attention Is All You Need,” 2017, Accessed: Jul. 11, 2024. [Online]. Available: https://www.semanticscholar.org/reader/204e3073870fae3d05bcbc2f6a8e263d9b72e776

[20] A. Juan, C. Vecino, H. Dirigido, M. Castro, and P. Madrid, “Diseño y creación de un LLM-ChatBot genérico-especializado,” 2024, Accessed: Jul. 11, 2024. [Online]. Available: https://repositorio.comillas.edu/xmlui/handle/11531/84378

[21] “meta-llama/Llama-3.2-1B · Cara abrazada.” Accessed: Jun. 10, 2025. [Online].

Available: https://huggingface.co/meta-llama/Llama-3.2-1B

[22] J. P. Crespo Obaco and J. Benavides Bailón, “Beneficios y desafíos de los asistentes virtuales en el aprendizaje,” *LATAM Revista Latinoamericana de Ciencias Sociales y Humanidades*, vol. 5, no. 2, Apr. 2024, doi: https://doi.org/10.56712/latam.v5i2.1909.

[23] Puma Quilumba William Geovanny, “Implementación de un chatbot como estrategia de apoyo en el servicio de soporte bibliotecario de la Universidad Técnica del Norte aplicando técnicas de procesamiento de lenguaje natural,” 2023. Accessed: Apr. 26, 2024. [Online]. Available: https://repositorio.utn.edu.ec/handle/123456789/14912

[24] A. Julián Morán Bastidas Director, “Implementación de un asistente virtual (CHATBOT) para el blog de la Carrera de Software de la Universidad Técnica del Norte utilizando inteligencia artificial,” 2023. Accessed: Apr. 26, 2024. [Online]. Available: https://repositorio.utn.edu.ec/handle/123456789/14731

[25] D. H. P. Cañar, J. D. G. Gómez, R. R. R. Paucha, L. E. S. Ordoñez, and M. P. Á. Robalino, “El uso de chatbots y asistentes virtuales en la educación: revisión de su impacto en la enseñanza y la evaluación del aprendizaje,” *Neosapiencia. Revista especializada en Ciencias de la Educación*, vol. 3, no. 1, pp. 153–164, Jul. 2025, doi: 10.64018/NEOSAPIENCIA.V3I1.29.

[26] Y. Zamora Varela and M. del C. Mendoza Encinas, “La Inteligencia artificial y el futuro de la educación superior:,” *Horizontes pedagógicos*, vol. 25, no. 1, pp. 1–13, Aug. 2023, doi: 10.33881/0123-8264.hop.25101.

[27] F. Vera, “INTEGRACIÓN DE LA INTELIGENCIA ARTIFICIAL EN LA EDUCACIÓN SUPERIOR: DESAFIOS Y OPORTUNIDADES,” *Revista electrónica transformar*, vol. Volumen 04, 2023.

[28] “TensorFlow.” Accessed: Jul. 10, 2024. [Online]. Available: https://www.tensorflow.org/?hl=es

[29] “PyTorch.” Accessed: Jul. 11, 2024. [Online]. Available: https://pytorch.org/

[30] “Hugging Face – The AI community building the future.” Accessed: Jul. 11, 2024. [Online]. Available: https://huggingface.co/

[31] “TensorBoard | TensorFlow.” Accessed: Jul. 11, 2024. [Online]. Available: https://www.tensorflow.org/tensorboard?hl=es-419

[32] “Unsloth Docs | Unsloth Documentation.” Accessed: Jul. 07, 2025. [Online]. Available:

https://docs.unsloth.ai/

[33] J. P. Obregon Faubla, “Desarrollo de un sistema de tiempo real para detección de plagio en exámenes académicos mediante estimación de pose utilizando YOLO-NAS, OpenCV y técnicas de aprendizaje automático : Sistema de alertas, interfaz de usuario e infraestructura en la nube.,” Oct. 2025, Accessed: Jan. 19, 2026. [Online]. Available: https://bibdigital.epn.edu.ec/handle/15000/26895

[34] U. Carlos, I. De Madrid, V. Galán, C. Tutora, and E. Castro Galán, “Aplicación de la Metodología CRISP-DM a un Proyecto de Minería de Datos en el Entorno Universitario,” 2015.

[35] ( Ncr and J. Clinton, “CRISP-DM 1.0 Step-by-step data mining guide,” 1999.

[36] I. Lopez-Gazpio, “Revisiting Challenges and Hazards in Large Language Model Evaluation,” *Procesamiento del Lenguaje Natural*, vol. 72, no. 0, pp. 15–30, Mar. 2024, doi: 10.26342/2024-72-1.

[37] I. Pikabea Mentxaka, “Revisión de modelos de aprendizaje en el ámbito de pregunta-respuesta,” Oct. 2022, Accessed: Jun. 29, 2024. [Online]. Available: http://addi.ehu.es/handle/10810/58115

[38] T. Fin De Máster, E. Serrano Fernández, and L. B. Molina, “Entrenamiento de modelos Deep Learning para descripción multilingüe de imágenes,” 2019, Accessed: Jun. 29, 2024. [Online]. Available: https://oa.upm.es/63705/

[39] C. Rafael, O. Lezcano, Y. Almeida, C. M. Suilán, and E. Velarde, “Generación Automática de Frases Equivalentes en Lenguaje Natural,” 2021, Accessed: Jun. 29, 2024. [Online]. Available: https://fototeca.uh.cu/s/scriptorium/item/2177053#lg=1&slide=0

[40] N. M. Gardazi, A. Daud, M. K. Malik, A. Bukhari, T. Alsahfi, and B. Alshemaimri, “BERT applications in natural language processing: a review,” *Artificial Intelligence Review 2025 58:6*, vol. 58, no. 6, pp. 1–49, Mar. 2025, doi: 10.1007/S10462-025-11162-5.

[41] P. Ignacio *et al.*, “ANÁLISIS DE LA PERFORMANCE Y HARDWARE DE LOS MODELOS LLMS DE GPT Y OPEN SOURCE APLICADOS A DIALOG SUMMARIZATION,” 2025. Accessed: Jan. 15, 2026. [Online]. Available: https://repositorio.uchile.cl/handle/2250/204989

[42] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, “BERTScore: Evaluating Text Generation with BERT,” *8th International Conference on Learning Representations,*

*ICLR 2020*, Apr. 2019, Accessed: Jun. 18, 2025. [Online]. Available: https://arxiv.org/pdf/1904.09675

[43] A. J. G. Megías, L. A. U. Lopez, and E. Martínez-Cámara, "The Influence of the Perplexity Score in the Detection of Machine-generated Texts," 2024. Accessed: Jan. 19, 2026. [Online]. Available: https://aclanthology.org/2024.nlpaics-1.10/

[44] J. J. Gilces Reyes, E. D. Castillo Suarez, and ESPOL.FIEC, "Desarrollo de un asistente virtual para mejorar el acceso a la información académica y administrativa de la FIEC - ESPOL.," 2023, Accessed: Apr. 26, 2024. [Online]. Available: http://www.dspace.espol.edu.ec/handle/123456789/60609

[45] K. K. Magallanes Ronquillo, A. J. Mora Rodríguez, J. F. Aguas Veloz, and L. del R. Plúas Pérez, "La inteligencia artificial aplicada en la innovación educativa en el proceso de enseñanza y aprendizaje," *LATAM Revista Latinoamericana de Ciencias Sociales y Humanidades*, vol. 4, no. 2, Jun. 2023, doi: 10.56712/latam.v4i2.706.

[46] A. Berggren, K. Remnelius, and Q. Dang, "Understanding Ambiguity: Designing, evaluating, and improving an LLM-powered virtual assistant for ambiguous task resolution in realistic scenarios," 2025, Accessed: Sep. 02, 2025. [Online]. Available: https://urn.kb.se/resolve?urn=urn:nbn:se:su:diva-244758

[47] B. H. Espinosa-Luna, J. Castillo-Oliva, B. A. Montañez-Díaz, and A. Mendoza-De-los-Santos, "Implementación de un chatbot basado en modelo de lenguaje de inteligencia artificial para responder preguntas frecuentes de estudiantes universitarios," *Revista Científica de Sistemas e Informática*, vol. 3, no. 2, p. e570, Jul. 2023, doi: 10.51252/rcsi.v3i2.570.

[48] B. Chen, C. Shu, E. Shareghi, N. Collier, K. Narasimhan, and S. Yao, "FI R EAC T : TOWARD LANGUAGE AGENT FINE-TUNING", Accessed: Jun. 09, 2025. [Online]. Available: https://fireact-agent.github.io.

[49] D. M. Anisuzzaman, J. G. Malins, P. A. Friedman, and Z. I. Attia, "Fine-Tuning Large Language Models for Specialized Use Cases," *Mayo Clinic Proceedings: Digital Health*, vol. 3, no. 1, p. 100184, Mar. 2025, doi: 10.1016/J.MCPDIG.2024.11.005.

[50] ----------Miguel-Ángel Cabeza-Rodríguez, "Asistentes ChatGPT en educación superior en línea y satisfacción del alumnado: un caso de estudio," *RIED-Revista Iberoamericana de Educación a Distancia*, vol. 28, no. 2, pp.

9–38, Apr. 2025, doi: 10.5944/RIED.28.2.43552.

[51] J. Stefania and E. Franco, “Inteligencia Artificial: Herramienta Dinámica en el Proceso de Enseñanza-Aprendizaje en la Educación Superior,” *Ciencia Latina Revista Científica Multidisciplinar*, vol. 9, no. 1, pp. 11824–11835, Mar. 2025, doi: 10.37811/CL_RCM.V9I1.16755.

[52] “meta-llama/Llama-3.2-3B · Hugging Face.” Accessed: Jul. 07, 2025. [Online]. Available: https://huggingface.co/meta-llama/Llama-3.2-3B?utm_source=chatgpt.com

[53] C. M. Santibáñez-Camarillo, C. E. Millán-Hernández, and E. Sánchez-Soto, “Evaluación de coherencia en respuestas generadas por un Chatbot para consulta de legislación y reglamentos universitarios,” *Publicación Semestral Pädi*, vol. 13, pp. 39–47, 2025, doi: 10.29057/icbi.v13iEspecial2.14676.

[54] J. P. A. Curi Garrafa, V. R. Ortega Marocho, and W. Mamani Rodrigo, “Comparación de modelos generativos compactos para respuesta automática en español mediante RAG,” *C&T Riqchary Revista de investigación en ciencia y tecnología*, vol. 7, no. 2, pp. 9–18, Aug. 2025, doi: 10.57166/riqchary.v7.n2.2025.2.

[55] J. Manuel, C. Recuero, and E. A. Cabrera, “MÉTODOS Y HERRAMIENTAS PARA LA EVALUACIÓN DE RESÚMENES AUTOMÁTICOS MEDIANTE FEEDBACK HUMANO TRABAJO DE FIN DE MÁSTER PRESENTADO POR DAVID CELESTINO COLÁS ROMANOS DIRIGIDO POR CONVOCATORIA DE SEPTIEMBRÉ,” 2023, Accessed: Jan. 14, 2026. [Online]. Available: https://hdl.handle.net/20.500.14468/14274

[56] D. Quiñones, C. Rusu, and V. Rusu, “A methodology to develop usability/user experience heuristics,” *Comput. Stand. Interfaces*, vol. 59, pp. 109–129, Aug. 2018, doi: 10.1016/J.CSI.2018.03.002.

[57] N. Ríos, A. Gustavo, J. López, A. Renata, Q. Contreras, and D. Felipe, “Uso de Chatbots en la enseñanza de Educación Superior ecuatoriana: una revisión sistemática de los modelos de estudios,” *ASCE MAGAZINE*, vol. 4, no. 3, pp. 2431–2453, Sep. 2025, doi: 10.70577/asce/2431.2453/2025.

[58] N. Segovia-García and A. Guzmán Rincón, “Evaluación de un chatbot basado en aprendizaje supervisado: impacto en la satisfacción y propuestas de mejora,” *Edutec, Revista Electrónica de Tecnología Educativa*, no. 92, pp. 234–252, Jun. 2025, doi:

10.21556/edutec.2025.92.3849.

[59] I. Dužević, T. Baković, and V. Surman, “Understanding artificial intelligence chatbot quality and experience: A higher education student perspective,” *Entrepreneurial Business and Economics Review*, vol. 13, no. 3, pp. 151–170, Sep. 2025, doi: 10.15678/EBER.2025.130308.

[60] C. De Software, B. Orlando De La, and C. Espinosa, “Desarrollo de un chatbot para la gestión del portafolio estudiantil basado en un modelo de lenguaje de gran escala (LLM) utilizando scrum y la ISO/IEC 25022,” Jan. 2026, Accessed: Jan. 26, 2026. [Online]. Available: https://repositorio.utn.edu.ec/handle/123456789/18478

[61] F. Vargas, A. González Coene, G. Escalante, E. Lobón, and M. Pulido, “impact of LLaMA fine tuning on hallucinations for name entity extraction in legal documents,” *SADIO Electronic Journal of Informatics and Operations Research*, vol. 24, no. 1, p. e068, Apr. 2025, doi: 10.24215/15146774e068.

# ANEXOS

**Anexo A. Repositorio Dataset de la Normativa**

GitHub: https://github.com/Dylanjesus65/FicaAsistant/tree/main/Dataset

**Anexo B. Repositorio con Documentación Del Entrenamiento del Modelo**

GitHub: https://github.com/Dylanjesus65/FicaAsistant/blob/main/ENTRENAMIENTO.md

**Anexo C. Repositorio con Documentación y Arquitectura Del Modelo**

GitHub: https://github.com/Dylanjesus65/FicaAsistant/blob/main/ARQUITECTURA.md

**Anexo D. Repositorio con Documentación y Evaluación del Modelo**

GitHub: https://github.com/Dylanjesus65/FicaAsistant/blob/main/METRICAS.md

**Anexo E. Proyecto completo de FicaAsistant**

GitHub: https://github.com/Dylanjesus65/FicaAsistant