

Chatbot basado en un Modelo Grande de Lenguaje para la atención al cliente



Chatbot based on Large Language Model for customer service

Balvir Victor Purewal Martinez  ORCID, Fany Yexenia Sobero Rodriguez ORCID

Universidad Nacional Mayor de San Marcos, Lima, Perú

Recibido: 27/12/2024 Revisado: 08/04/2025 Aceptado: 15/05/2025 Publicado: 30/06/2025

Resumen

Los chatbots basados en modelos como BERT debían actualizarse cada vez que se agregaban nuevas conversaciones, lo que limitaba su flexibilidad. Esto propició el desarrollo de chatbots con botones preconfigurados, restringiendo a los usuarios que preferían texto libre. Otras soluciones, como RASA, incorporaron módulos de Comprensión del Lenguaje Natural para mejorar la interpretación de las intenciones del usuario y módulos de acciones para recuperar respuestas. También implementaron un servicio de consulta de información usando una matriz de términos del documento y similitud de coseno, lo que permitía acceder a más datos y generar diálogos más fluidos. Sin embargo, configurar estas soluciones era complejo, ya que implicaba considerar todos los posibles escenarios. Para abordar estas limitaciones, esta investigación propone un chatbot que integre tecnologías modernas, como bases de datos vectoriales, el modelo GPT-3.5-turbo y la generación aumentada por recuperación. La evaluación mostró una precisión de 0.98, sensibilidad de 1.00 y puntuación F1 de 0.99, lo que demuestra su eficacia en la atención al cliente. Esta estrategia permite reducir tiempos y costos, y generar respuestas más eficientes para los usuarios en las redes sociales.

Palabras Clave: Chatbot, modelos grandes de lenguaje, generación aumentada por recuperación, base de datos vectorial

Abstract

Chatbots based on models like BERT needed updating every time new conversations were added, which limited their flexibility. This led to the development of chatbots with pre-configured buttons, restricting users who preferred free text. Other solutions, like RASA, incorporated Natural Language Understanding modules to improve the interpretation of user intentions and action modules to retrieve responses. They also implemented an information retrieval service using a document term matrix and cosine similarity, enabling access to more data and generating more fluid dialogues. However, configuring these solutions was complex, as it involved considering all possible scenarios. To address these limitations, this research proposes a chatbot that integrates modern technologies like vector databases, the GPT-3.5-turbo model, and retrieval-augmented generation. The evaluation showed a precision of 0.98, a sensitivity of 1.00, and an F1-score of 0.99, demonstrating its effectiveness in customer service. This strategy allows for reduced time and costs and generates more efficient responses for users on social networks.

Keywords: Chatbot, large language model, retrieval augmented generation, vector database

INTRODUCCIÓN

Durante los procesos de admisión a programas de maestría y doctorado, la universidad enfrenta una problemática recurrente. La gran cantidad de usuarios que solicitan información sobre los requisitos, los plazos y los pasos a seguir para aplicar a estos programas, supera la capacidad del personal administrativo para atenderlos de manera oportuna y satisfactoria. Esto ocasiona retrasos, reiteraciones de consultas e insatisfacción por parte de los postulantes, quienes requieren una respuesta rápida y precisa a sus inquietudes.

Los chatbots se han convertido en una tecnología crucial para la interacción y atención al cliente, ofreciendo diálogos naturales que proporcionan información, resuelven dudas y realizan acciones en nombre del usuario. Esta tecnología ha demostrado ser muy útil para optimizar la experiencia del cliente, ya que pueden ofrecer un servicio rápido, eficiente y disponible las 24 horas del día (Khennouche et al., 2023). Su implementación se ha expandido rápidamente en diversos sectores, incluyendo la educación superior (Cordero et al., 2020), donde las instituciones buscan automatizar y optimizar los procesos de servicio, reducir costos operativos y mejorar la satisfacción de los usuarios (Nichifor et al., 2021).

El estudio de Haugeland et al. (2022) revela una discrepancia entre las expectativas del usuario y el rendimiento real de la entrada de texto libre en chatbots. Si bien se anticipaba que el texto libre brindaría mayor flexibilidad, las limitaciones en el procesamiento del lenguaje natural del chatbot obstaculizaron esta funcionalidad. El chatbot frecuentemente malinterpretó o no comprendió las entradas de texto libre, resultando en una experiencia no más flexible que la interacción con botones predefinidos. Esta limitación tecnológica frustró a los usuarios que deseaban una opción de texto libre genuinamente adaptable a sus expresiones naturales, exponiendo la necesidad de mayor sofisticación en la comprensión del lenguaje natural para futuras implementaciones de chatbots. Selamat y Windarasi (2021) sugieren como línea de trabajo futura el uso de técnicas más avanzadas de procesamiento de lenguaje natural (NLP) con el objetivo de desarrollar chatbots capaces de mantener conversaciones más naturales e informales, similares a las interacciones humanas.

Hay un estudio donde se introduce la arquitectura Transformer, un nuevo modelo de redes neuronales basado completamente en mecanismos de atención, eliminando la necesidad de recurrencia y convoluciones. Demuestra que este modelo es superior en calidad y más paralelizable, requiriendo menos tiempo de entrenamiento en tareas de traducción automática. El Transformer utiliza atención multi-cabeza en tres formas: atención encoder-decoder, autoatención en el encoder y autoatención en el decoder. (Vaswani et al., 2017) Como su nombre lo indica, Transformador Generativo Preentrenado (GPT) se pre-entrena en un corpus masivo de texto sin etiquetar. Durante el pre-entrenamiento, el modelo aprende a predecir la siguiente palabra en una secuencia dada una secuencia de palabras anteriores. Este proceso permite que el modelo desarrolle una comprensión profunda del lenguaje, incluyendo la gramática, la semántica y el razonamiento. Después del pre-entrenamiento, GPT se puede ajustar a una variedad de tareas específicas, como la traducción automática, la generación de texto y la respuesta a preguntas (Brown et al., 2020).

Desde el GPT original, ha habido varias iteraciones, incluyendo GPT-2 y GPT-3. Estas versiones posteriores han aumentado significativamente el tamaño del modelo y la cantidad de datos de entrenamiento, lo que ha llevado a mejoras significativas en el rendimiento. Por ejemplo, GPT-3 tiene 175 mil millones de parámetros, lo que le permite generar texto de alta calidad y realizar una variedad de tareas con un rendimiento impresionante. (Fan et al., 2023)

Debido a los avances tecnológicos, hoy se dispone de chatbots más avanzados, impulsados por modelos de procesamiento de lenguaje natural como ChatGPT. Estos modelos sobresalen por su capacidad para realizar tareas conversacionales con mayor eficiencia, mejor comprensión del contexto, generación de respuestas más precisas y mayor coherencia. A diferencia de los modelos de Redes Generativas Antagónicas (GAN), ChatGPT se basa en la arquitectura de Transformador Generativo Preentrenado (GPT), lo que le permite ofrecer interacciones más naturales y fluidas (Ray, 2023). A pesar de su impresionante rendimiento, GPT tiene algunas limitaciones. Por ejemplo, puede generar texto que es factualmente incorrecto o que contiene sesgos presentes en los datos de entrenamiento. Además, el tamaño masivo de los modelos GPT los hace computacionalmente costosos de entrenar y ejecutar. (Suresh & Shunmugapriya, 2024) (Packer et al., 2024)

Asimismo, estos modelos de lenguaje generativo, como GPT-3.5, han impulsado significativamente la capacidad de los chatbots para lograr interacciones más naturales, fluidas y contextualizadas. Estos modelos de lenguaje de gran tamaño han demostrado una mayor comprensión del lenguaje natural, permitiendo a los chatbots mantener diálogos más coherentes y adaptados al contexto específico de cada consulta. Esta mejora en la capacidad conversacional ha consolidado a los chatbots como herramientas valiosas para optimizar la experiencia de los usuarios, ofreciendo respuestas más relevantes y personalizadas (Khennouche et al., 2023) (Shahsavari & Choudhury, 2023). Sin embargo, la aplicación de chatbots en el ámbito universitario presenta desafíos específicos. Si bien ofrecen un gran potencial para la atención automatizada, estudios han revelado limitaciones en la precisión y contextualización de las respuestas, especialmente al abordar consultas complejas o específicas de la institución (Mageira et al., 2022).

El objetivo de la investigación es explorar ¿cómo implementar un chatbot para asistir las necesidades de información en un entorno universitario? Para ello es necesario proponer soluciones que se adapten a contextos donde el usuario tiene la necesidad de un chatbot que le brinde una experiencia de texto libre con mayor sofisticación en la comprensión del lenguaje natural.

El trabajo comienza con una introducción a los chatbots, explicando su evolución y cómo la arquitectura de los Transformer y los Transformer Generativo Preentrenado ha contribuido a su desarrollo. Además, proporciona antecedentes del problema abordado, seguido de los materiales y métodos utilizados para su estudio, y posteriormente describe los resultados obtenidos. Finalmente, se incluye una breve discusión sobre los trabajos futuros y las conclusiones.

ANTECEDENTES DEL PROBLEMA

En el estudio de Lewis et al. (2021), se propone una metodología de ajuste fino de propósito general para la generación aumentada por recuperación (RAG, por sus siglas en inglés), que integra tanto memoria paramétrica como no paramétrica preentrenado para la generación de lenguaje. Los modelos RAG presentados en su trabajo combinan un modelo seq2seq preentrenado como memoria paramétrica, con un índice vectorial denso de Wikipedia como memoria no paramétrica, el cual es accesible mediante un recuperador neuronal también preentrenado. Esta combinación permite una mejora en el acceso y manipulación de la información, lo que a su vez potencia la generación de respuestas más precisas y contextualizadas.

Este trabajo describe el desarrollo de un prototipo de chatbot utilizando RASA para abordar las necesidades de información relacionadas con la pandemia del COVID-19. La herramienta se diseñó y construyó con RASA, incluyendo módulos de Comprensión del Lenguaje Natural para mejorar el entendimiento de las intenciones, y un módulo de acciones que permite realizar la búsqueda de respuestas. Además, se implementó un servicio de consulta de información utilizando una matriz de términos del documento y similitud de coseno, lo que permite acceder a más información y generar conversaciones más fluidas. El modelo resultante respondió de manera eficaz a preguntas sobre el escenario de riesgo de la pandemia. El estudio exploró las herramientas existentes para implementar un chatbot, seleccionó la más adecuada, y construyó conjuntos de datos para el escenario de la pandemia. Integrar diversas fuentes de datos heterogéneas con grandes volúmenes de información fue un desafío, al igual que diseñar la arquitectura de procesamiento del lenguaje natural para modelar y entrenar los modelos. El modelo de Procesamiento del Lenguaje Natural más grande mostró valores superiores al 88% en precisión, sensibilidad y puntaje F1. El prototipo de chatbot mantuvo 1480 conversaciones con un promedio de participación de 4.12 minutos. El 87% de los encuestados indicó que volvería a utilizar el chatbot para obtener información sobre la COVID-19. Los resultados son prometedores y demuestran que el diseño de la arquitectura y el corpus construido permiten desarrollar un chatbot para escenarios de pandemia. (Peña-Torres et al., 2022)

Este trabajo presenta un Edu-Chatbot altamente adaptable, desarrollado con el framework Xatkit, para mejorar el apoyo a estudiantes. El chatbot utiliza un framework codificador-decodificador, procesamiento de lenguaje natural y aprendizaje profundo. Incorpora reconocimiento de intenciones con un motor PNL basado en CamemBERT, reconocimiento y síntesis de voz, y un avatar 3D para una interacción más atractiva. La arquitectura del sistema procesa la entrada del usuario (texto o audio) a través del reconocedor de voz y el motor PNL para identificar la intención. Las intenciones reconocidas activan acciones y respuestas, entregadas por el avatar 3D con voz y gestos sincronizados. La conclusión principal es que el sistema propuesto, con su arquitectura y componentes (como el motor NLP basado en CamemBERT, el reconocimiento y síntesis de voz, y el avatar 3D), tiene el *potencial* de ser una herramienta útil para el apoyo a los estudiantes. (Baha et al., 2022)

En su estudio, Peyton y Unnikrishnan (2023) implementaron un modelo basado en BERT llamado SBERT para tareas de búsqueda semántica y clasificación de intenciones, destacando su efectividad en comparación con plataformas más accesibles como Dialogflow y QnA Maker. Sin embargo, la implementación de SBERT

presenta varias complejidades: (1) requiere ajuste fino específico para cada tarea, lo cual demanda un proceso más técnico que las soluciones preconfiguradas, (2) involucra el uso de infraestructura adicional, como Elasticsearch, para almacenar las incrustaciones y realizar búsquedas rápidas, lo que agrega una capa de complejidad en la configuración y mantenimiento, (3) exige scripts personalizados para transformar y procesar los datos, lo que limita la automatización, y (4) requiere una evaluación manual del rendimiento del modelo, lo cual puede ser más tedioso y laborioso. Estos aspectos hacen que la implementación de SBERT sea menos accesible y más difícil de manejar en comparación con otras plataformas más intuitivas.

Dongbo et al. (2023) proponen un sistema de chatbot inteligente que utiliza un enfoque híbrido de aprendizaje automático, combinando Redes Neuronales Bidireccionales de Memoria Larga a Corto Plazo y un clasificador Fuzzy Naive Bayes, para el análisis de sentimientos. Este sistema mejora la interacción con el usuario al detectar la polaridad emocional del texto y generar respuestas contextualmente relevantes mediante un modelo Secuencia a Secuencia (Seq2Seq). El algoritmo propuesto obtuvo los siguientes resultados: con Bag of Words y análisis de sentimiento, exactitud: 0.927, precisión: 0.933, recall: 0.901, F1-Score: 0.915; con TF-IDF y análisis de sentimiento, exactitud: 0.925, precisión: 0.931, recall: 0.911, F1-Score: 0.935. La evaluación del sistema en los dominios de películas y restaurantes, utilizando métricas como precisión, recall y F-measure, demuestra una mejora significativa en la capacidad del chatbot para comprender y responder a las emociones de los usuarios.

MATERIALES Y MÉTODOS

La metodología para utilizar es SCRUM para ello se definió una ruta de implementación como la figura 1 y después se desarrollo el backlog con las historias de usuario como la figura 2.



Figura 1. Metodología con ruta de implementación



Figura 2. Backlog

De acuerdo con la metodología SCRUM, en la etapa inicial del proyecto (sprint 0) se llevó a cabo la fase de concepción, la cual incluyó el análisis de los interesados y el desarrollo de un diagrama de arquitectura preliminar. El diagrama adjunto en la figura 3 es el resultado final del proyecto implementado.

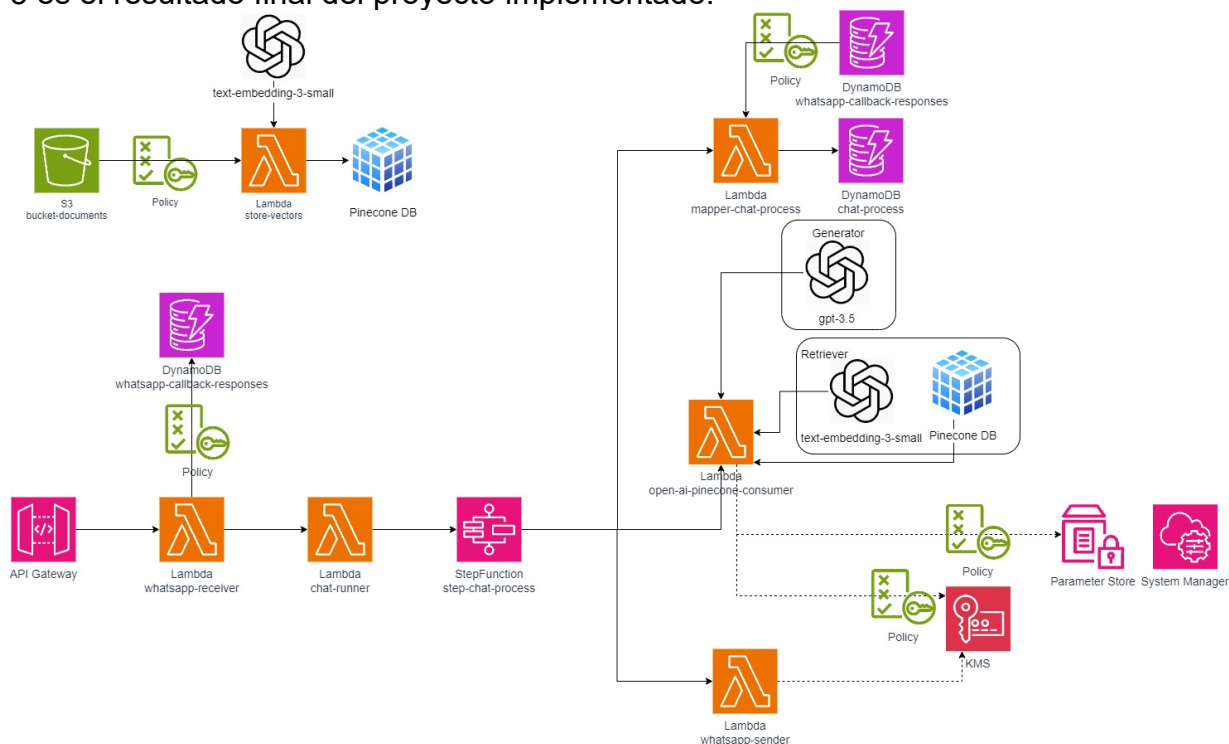


Figura 3. Diagrama de arquitectura del chatbot

Descripción de componentes de la arquitectura serverless (sin servidores) en AWS

- La lambda open-ai-pinecone-consumer tiene el generador (Generator) que consume al LLM (GPT-3.5-turbo versión optimizada para rendimiento y

eficiencia) y el recuperador (Retriever) que consume la base de datos vectorial serverless de Pinecone con un algoritmo de recuperación de documentos y utiliza el modelo text-embedding-small-3 para decodificar la información. El generador finalmente retorna la respuesta más apropiada con la data del contexto y la data del vector que se relacione más al tema.

- La lambda-store-vectors consume el modelo text-embedding-small-3 y la base de datos vectorial para generar y almacenar los vectores.
- El endpoint que expongo con el Api gateway es para que el API de whatsapp haga el callback debido a que Meta Bussines al crear la cuenta dice que debe ser implementado con un callback para recibir los mensajes de whatsapp.
- La lambda whatsapp-receiver obtiene los datos de los mensajes que envían los usuarios.
- La lambda chat runner verifica el tipo de caso y dispara la step-chat-process encargada de ejecutar el flujo de respuesta.
- La lambda mapper-chat-process se encarga de mapear cada paso del step function, además de obtener y almacenar datos con las bases de datos de DynamoDb.
- La lambda-whatsapp-sender responsable de consumir el api de whatsapp para responder el mensaje basada en la respuesta.
- El bucket para almacenar los documentos en formato Markdown que al ser subidos, se convertirán en vectores y subirán a pinecone en formato de vectores. Ese S3 dispara un trigger que ejecuta una lambda encargada de consumir un API de open-ai con el modelo text-embeddings-small-3 para generar los vectores y otra API para almacenar los vectores en Pinecone.
- El parameter store con el KMS para almacenar los secretos encriptados.

Sprint 1: En la primera etapa se configuro el modelo grande de lenguaje y se agregó memoria a corto plazo.

Informe sobre el desarrollo de la Historia de Usuario 1: Conexión y configuración del contexto del modelo de lenguaje de gran tamaño.

Se implementó la conexión y configuración del modelo grande de lenguaje (GPT-3.5-turbo) de OpenAI, además se definió la interacción inicial del chatbot con el usuario. Se realizo una conversación por terminal como se visualiza en la figura 4, en la cual el chatbot se presenta como un asistente virtual de la Facultad de Ingeniería de Sistemas e Informática, especializado en brindar información sobre los programas de postgrado.

```
Tú: hola
Asistente virtual: ¡Hola! Bienvenido a la Facultad de Ingeniería de Sistemas e Informática de la UNMSM. ¿En qué puedo ayudarte hoy?
Tú: que programas ofrecen
Asistente virtual: ¡Hola! Soy el Asistente Virtual de la Facultad de Ingeniería de Sistemas e Informática de la UNMSM. En la unidad de postgrado, ofrecemos los siguientes programas:

1. Doctorado en Ingeniería de Sistemas e Informática.
2. Maestría en Ingeniería de Sistemas e Informática con mención en Gestión de Tecnología de Información y Comunicaciones.
3. Maestría en Ingeniería de Sistemas e Informática con mención en Ingeniería de Software.
4. Maestría en Gobierno de Tecnologías de Información.
5. Maestría en Gestión de la Información y del Conocimiento.

¿Te gustaría saber más detalles sobre alguno de estos programas? ¡Estoy aquí para ayudarte!
Tú: █
```

Figura 4. Conversación por terminal

Informe del desarrollo de la Historia de Usuario 2: Creación de memoria a corto plazo para el chatbot.

Se integro la base de datos no relacional (DynamoDB) para almacenar el historial de la conversación y un contador de tokens con el fin de conocer la cantidad que se están utilizando por cada interacción. Se implementó una función para calcular el número de tokens a partir de una cadena de texto, utilizando la biblioteca Tiktoken. Adicionalmente se modificó la lógica de obtención del historial para ser parametrizable en base a la cantidad de tokens, y que traiga las últimas conversaciones en base a K parámetro configurable con cantidad de tokens actualmente configurado en 4098 tokens. El método `get_answer` se encarga de procesar la entrada del usuario, agregar los mensajes al historial de la conversación y generar la respuesta del chatbot como se visualiza en la figura 5. Se utiliza el modelo de lenguaje GPT-3.5-turbo de OpenAI para producir las respuestas.

```
def _trim_memory(self, chat_history: list) -> list:
    total_tokens = 0
    trimmed_history = []

    for message in reversed(chat_history):
        message_tokens = self._calculate_token_count(message["content"])
        if total_tokens + message_tokens <= self.token_limit:
            trimmed_history.insert(0, message)
            total_tokens += message_tokens
        else:
            break

    formatted_history = [{"role": msg["role"], "content": msg["content"]} for msg in trimmed_history]

    return formatted_history

def get_answer(self, question: str, phone: str) -> str:
    existing_messages = self.load_messages(phone)
    self.save_message("human", question, phone)

    if not existing_messages:
        self.memory.chat_memory.messages = []

    chat_history = self.load_messages(phone)
    trimmed_history = self._trim_memory(chat_history)
    assistant_response = result["answer"]

    logging.info(f"Assistant response: {assistant_response}")

    self.save_message("assistant", assistant_response, phone)

    return assistant_response
```

Figura 5. Código de memoria a corto plazo con conteo de tokens

Retrospectiva sobre el Sprint 1:

El uso de memoria a corto plazo presenta limitaciones, ya que a medida que la conversación se prolonga, aumenta el consumo de tokens, como se evidencia en el ejemplo con 390 tokens. Si bien se cumple el objetivo, el costo computacional se incrementa a medida que se utiliza más la conversación. Otro desafío identificado es que la información disponible para responder se concentra en el contexto proporcionado a través del atributo "system" del modelo de OpenAI. Además, se observa que en la última parte de la conversación, el chatbot proporcionó datos sobre el envío y el carácter de plataforma de comercio electrónico, lo cual se conoce como "alucinación", un fenómeno que debe ser abordado.

Sprint 2: En la segunda fase del proyecto, se realizó la recopilación, procesamiento de datos, así configuración de la base de datos vectorial y generación de vectores.

Informe del desarrollo de la Historia de Usuario 3: Recopilación y procesamiento de datos

En esta fase se recopilaron los datos, los cuales fueron enviados por la facultad como se evidencia en la figura 6. Adicionalmente estos formatos fueron convertidos a formato Markdown como se visualiza en la figura 7.



Figura 6. Evidencia de recopilación de datos

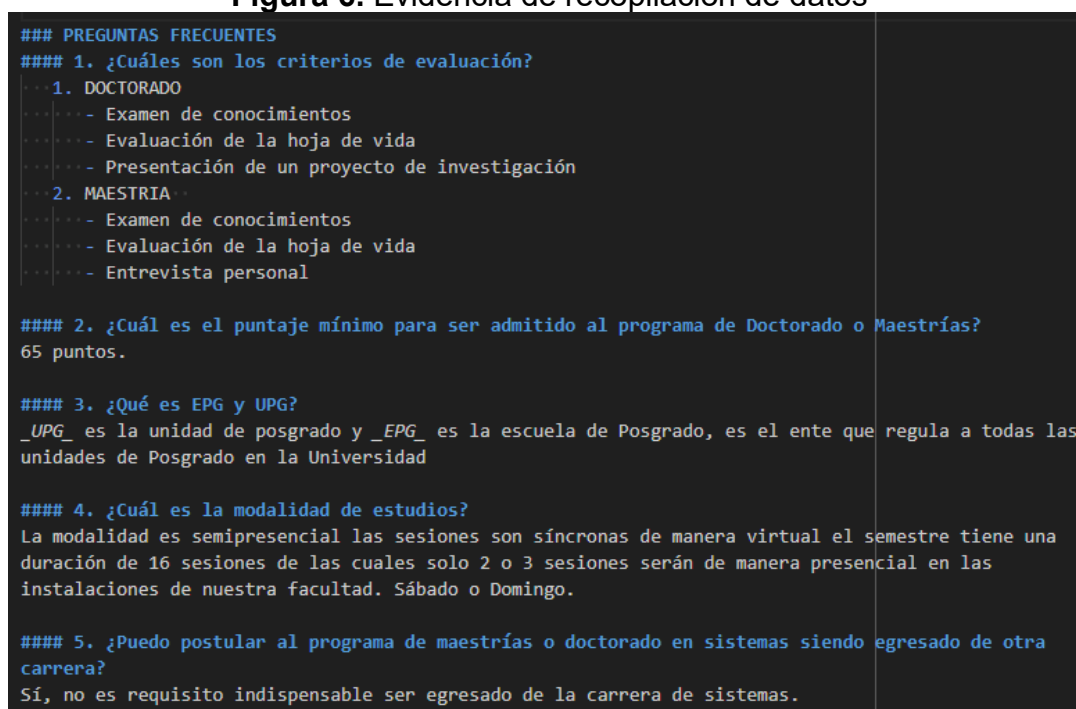


Figura 7. Evidencia del procesamiento de datos, preguntas frecuentes en formato Markdown

Informe del desarrollo de la Historia de Usuario 4: Configuración de la DB vectorial

Se implementó la configuración y el proceso de almacenamiento de los datos en una base de datos vectorial. Esto incluye el uso de la biblioteca Pinecone para crear el índice, definir el espacio de nombres y almacenar los vectores de texto, el uso de un modelo de menos dimensiones que servirá como recuperador en este caso “text-embedding-small-3”. Además, se utiliza la biblioteca Marker para realizar el procesamiento de los datos (la conversión de los PDFs a formato Markdown), y MarkdownHeaderTextSplitter de la biblioteca Langchain para la división del contenido en formato Markdown basados en los Header que presente el documento, de esa

forma hacemos que nuestros textos utilizados para generar los vectores sean de tamaños apropiados y no contengan tanta información. Se muestra la evidencia en la figura 8 de que los vectores han sido almacenados correctamente en la base de datos vectorial.

Informe sobre el desarrollo de la Historia de Usuario 5: Creación de vectores

Matches: 5

1	ID	VALUES
	5ccdf532-ec...	-0.0109714391, 0.049615778, 0.0478834435, -0.0218762495, -0.00151856709, -0.0078787813, -0.0142362192, -0.00421700813, -0.011693243...
SCORE	METADATA	
0.0330	: "Contenido de los cursos de la Maestría En Ingeniería De Sistemas E Informática Con Mención En Software"	
	text: "### VIII. Sumillas \n#### Asignaturas Obligatorias: Machine Learning Big Data \nAsignatura que corresponde al periodo Profundización, es de naturaleza Teórica-practica de modalidad...	
2	ID	VALUES
	d526321b-a2...	-0.0178899262, 0.0461703576, 0.0488658026, -0.0311497748, 0.00187757274, -0.00446704729, -0.0128359683, 0.00136945979, -0.003654...
SCORE	METADATA	
0.0290	: "Contenido de los cursos de la Maestría En Ingeniería De Sistemas E Informática Con Mención En Gestión De Tecnología De Información Y Comunicaciones"	
	text: "### VIII. Sumillas \n#### Habilidades Directivas En Gestión \nLa Asignatura corresponde al periodo de Profundización es de naturaleza teórica-practica y de modalidad presencial. TL...	
3	ID	VALUES
	d7c94bf1-76...	-0.0316196606, 0.0354284868, 0.0410247, 0.00997955631, -0.0191079769, 0.0146820759, 0.00522118, 0.0308749173, -0.022895528, -0.0054...
SCORE	METADATA	
0.0207	: "# REQUISITOS PARA POSTULAR A LOS PROGRAMAS DE DOCTORADO Y MAESTRIA \n#### DOCTORADO: \n1. Resumen de la hoja de vida del postulante (documentado, foliado y orde...	
4	ID	VALUES
	138963ba-3...	0.0195507091, 0.0445632748, 0.0384158306, -0.0316742063, -0.0230700653, -0.0143630868, 0.0211618431, 0.0243841112, -0.0449517779, -...
SCORE	METADATA	
0.0049	: "Datos generales de la Maestría En Ingeniería De Sistemas E Informática Con Mención En Software"	
	text: "### I. Presentación \nLa Maestría en Ingeniería de Sistemas e Informática con mención en Ingeniería de Software en la modalidad presencial, se crea en el año 2002, con el propósito...	

Figura 8. Evidencia del proceso de inserción de vectores:

Retrospectiva sprint 2:

En esta etapa se procesaron los datos al convertir los formatos PDFs a Markdown, además de utilizar un divisor de contenido para que los vectores que generemos sean de tamaños apropiados. Y el uso del modelo text-embedding-small-3 para transformar los datos en vectores y la base de datos vectorial de Pinecone para almacenar los vectores.

Sprint 3: Durante la tercera etapa del proyecto, se implementó un mecanismo de memoria a largo plazo, permitiendo al chatbot acceder a más contexto sin consumir demasiados tokens. Este mecanismo utiliza un modelo de embeddings que tiene menos dimensiones como recuperador, lo que permite acceder a la información de manera eficaz y evitar el consumo excesivo de tokens durante la conversación, junto con un algoritmo de búsqueda que trae los vectores más relacionados a la consulta. De esta forma, el chatbot puede mantener un contexto más amplio y proporcionar respuestas más coherentes y relevantes para los usuarios.

Retrospectiva sprint 3:

En esta etapa el chatbot es capaz de traer información relacionada a las consultas, además mantiene una conversación coherente. La implementación del modelo recuperador que decodifica con el modelo de menos dimensiones los contenidos de la base de datos vectorial lo que brinda acceso a más información en el contexto sin desperdiciar los tokens, debido a ajustes en el system del modelo generador de respuesta es capaz de responder que no cuenta con información cuando se consulta sobre algo que no esté relacionado a las consultas.

Sprint 4: En la cuarta etapa del proyecto, se implementó la integración con WhatsApp utilizando la API de WhatsApp Business. Esto implicó configurar una cuenta de Meta

Business y webhooks para recibir y responder mensajes. Se agregó lógica a los componentes whatsapp-receiver y whatsapp-sender para procesar consultas y enviar respuestas a la API de WhatsApp. Las pruebas con un conjunto de datos de 130 preguntas arrojaron 127 respuestas correctas, con las 3 restantes incorrectas debido a información faltante. La alta precisión, sensibilidad y puntuación F1 validaron el rendimiento del chatbot. Pruebas posteriores, después de corregir la información de origen, dieron como resultado respuestas precisas, lo que demuestra la capacidad de mejora del chatbot.

Retrospectiva Sprint 4:

Se evidencia la utilidad del chatbot al poder responder preguntas de distintos usuarios como se visualiza en la figura 9, se brinda atención concurrente y operatividad las 24 horas con una precisión alta sin necesidad de reentrenamiento. Adicionalmente se pudo observar los tiempos de respuesta y costos.

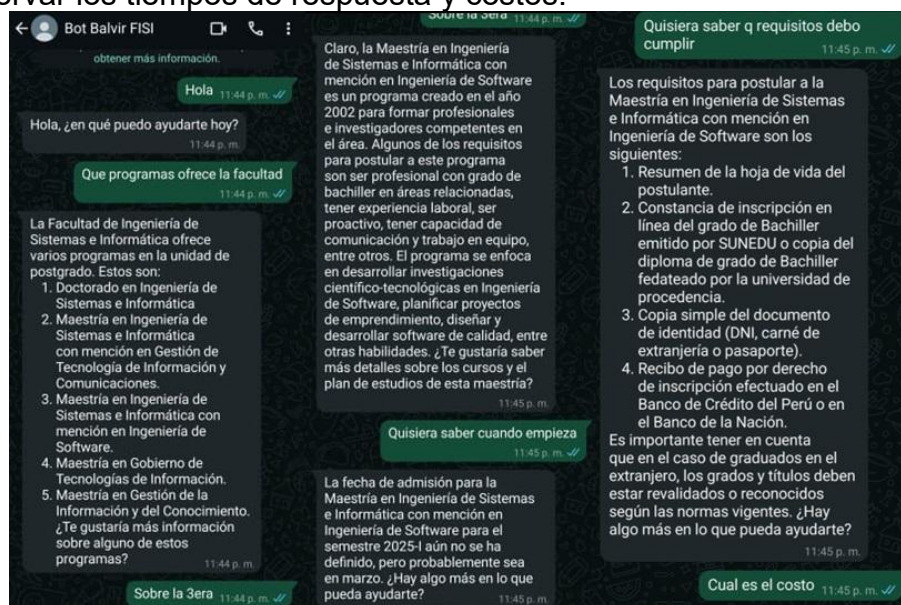


Figura 9. Conversación de prueba con usuario.

RESULTADOS

Se elaboró un dataset de 130 preguntas y respuestas, y obtuvieron los resultados de la tabla 1.

Tabla 1. Matriz de confusión

Matriz de confusión		Nivel predicho	
		NO	NO
Nivel actual	SI	Verdaderos Positivos: 127	Falsos Negativos: 0
	SI	Falsos Positivos: 3	Verdaderos Negativos: 0

Precisión (precision): $VP / (VP + FP) = 127 / (127 + 3) = 127 / 130 = 0.98$

Sensibilidad (recall): $VP / (VP + FN) = 127 / (127 + 0) = 127 / 127 = 1.00$

F1 score: $(2 * \text{Precisión} * \text{Sensibilidad}) / (\text{Precisión} + \text{Sensibilidad}) = (2 * 0.98 * 1.00) / (0.98 + 1.00) = 1.96 / 1.98 = 0.99$

El chatbot obtuvo una precisión de 0.98, una sensibilidad de 1.0 y una F1 score de 0.99.

Tabla 2. Comparativa con el modelo de Peña-Torres et al. (2022)

Modelo	Sensibilidad	Precisión	F1 Score
BC	0.86	0.86	0.85
BCN	0.87	0.88	0.86

BFAQ	0.89	0.89	0.89
Chatbot LLM con RAG	1.00	0.98	0.99

La tabla 2 compara nuestras métricas con el modelo de Peña-Torres et al. (2022) redondeado a 2 decimales, el chatbot tiene un alto desempeño en la generación de respuestas relevantes a las consultas, lo cual es esencial para brindar una excelente atención al cliente.

El chatbot ha demostrado responder de forma casi inmediata a las consultas del usuario.

La arquitectura serverless (sin servidores) permite tener un costo por uso. A la fecha se han gastado 1.53 dólares en el API de OpenAI, 0.07 céntimos de dólar en WhatsApp API y 0.02 céntimos de dólar en Amazon Web Services. Dando un total de 1.62 dólares. El costo promedio de una conversación con un usuario es 0.27 céntimos de dólar.

DISCUSIÓN

Los resultados obtenidos en este proyecto muestran que es posible desarrollar un chatbot basado en un modelo de lenguaje de gran tamaño que pueda brindar una atención al cliente de alta calidad al implementar mecanismos de memoria a corto y largo plazo, el chatbot es capaz de mantener el contexto de la conversación y acceder a información relevante, lo que le permite generar respuestas coherentes y precisas.

Con los resultados obtenidos con investigaciones previas en el campo (Goodman et al., 2023; Peña-Torres et al., 2022; Sulaiman, 2022), se puede observar que el enfoque adoptado en este proyecto ha logrado niveles de precisión y sensibilidad elevados, superando o igualando el desempeño de otros chatbots reportados.

En el futuro, se podría explorar la posibilidad de incorporar capacidades de análisis de sentimiento o emociones, con el fin de mejorar aún más la experiencia del usuario y brindar una atención más personalizada. También se podría realizar mediciones con usuarios finales, para validar la efectividad del chatbot en un entorno real.

Adicionalmente, sería interesante evaluar el rendimiento y la adaptabilidad del chatbot en diferentes dominios o contextos más amplios. Esto permitiría validar su escalabilidad y su capacidad para responder eficazmente a consultas en diversos ámbitos, más allá del dominio específico en el que fue desarrollado inicialmente. Al probar el chatbot en nuevos escenarios, se podría determinar su flexibilidad y su habilidad para mantener un alto nivel de desempeño y relevancia en las respuestas, lo que sería fundamental para garantizar una excelente atención al cliente en diferentes sectores o situaciones.

CONCLUSIONES

El desarrollo de un chatbot basado en un modelo grande de lenguaje como generador, con un modelo de menos dimensiones como el recuperador ha demostrado ser una solución efectiva para mejorar la atención al cliente.

A través de las diferentes fases del proyecto, se logró implementar un chatbot capaz de mantener conversaciones coherentes y proporcionar información relevante a los usuarios.

Este estudio describe el desarrollo de un chatbot para atención al cliente, especializado en información sobre programas de estudio de postgrado. El proyecto se dividió en cuatro etapas:

1. **Conexión y configuración:** Se configuró el modelo de lenguaje de OpenAI y se estableció la interacción inicial del chatbot, presentándose como un asistente virtual. Se implementó la memoria a corto plazo para el historial de conversación.
2. **Procesamiento de datos y base de datos vectorial:** Se recopilaron y procesaron datos, generando preguntas frecuentes en formato Markdown. Se configuró una base de datos vectorial con Pinecone y Langchain para almacenar y consultar información. Se generaron vectores de texto mediante un modelo recuperador de embeddings.
3. **Memoria a largo plazo:** Se implementó un mecanismo de memoria a largo plazo, lo que permite que el chatbot acceda a un contexto más amplio sin consumir excesivos tokens. Esto se logró utilizando un modelo recuperador de embeddings con un número reducido de dimensiones como recuperador, lo que facilita un acceso eficiente a la información.
4. **Conexión con WhatsApp:** Se integró el chatbot con la plataforma de WhatsApp, lo que permite a los usuarios interactuar con el asistente virtual a través de este canal de comunicación.

Las pruebas mostraron que el chatbot mantiene conversaciones coherentes, responde preguntas relevantes y maneja consultas fuera del dominio. La evaluación del rendimiento obtuvo una precisión de 0.98, sensibilidad de 1.00 y puntuación F1 de 0.99, demostrando su eficacia en la atención al cliente.

También se demostró que los tiempos de respuestas son muy bajos al tener respuestas inmediatas para los usuarios.

Finalmente se demostró que los costos operativos son bajos debido a la arquitectura propuesta, que permite un costo por uso.

REFERENCIAS

- Baha, T. A., Hajji, M. E., Es-saady, Y., & Fadili, H. (2022). Towards highly adaptive Edu-Chatbot. *Procedia Computer Science*, 198, 397-403. <https://doi.org/10.1016/j.procs.2021.12.260>
- Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., ... Amodei, D. (2020). *Language Models are Few-Shot Learners* (No. arXiv:2005.14165). arXiv. <https://doi.org/10.48550/arXiv.2005.14165>
- Cordero, J., Toledo, A., Guamán, F., & Barba-Guamán, L. (2020). Use of chatbots for user service in higher education institutions. *2020 15th Iberian Conference on Information Systems and Technologies (CISTI)*, 1-6. <https://doi.org/10.23919/CISTI49556.2020.9141108>

- Dongbo, M., Miniaoui, S., Fen, L., Althubiti, S. A., & Alsenani, T. R. (2023). Intelligent chatbot interaction system capable for sentimental analysis using hybrid machine learning algorithms. *Information Processing & Management*, 60(5), 103440. <https://doi.org/10.1016/j.ipm.2023.103440>
- Fan, L., Li, L., Ma, Z., Lee, S., Yu, H., & Hemphill, L. (2023). *A Bibliometric Review of Large Language Models Research from 2017 to 2023* (No. arXiv:2304.02020). arXiv. <https://doi.org/10.48550/arXiv.2304.02020>
- Goodman, R. S., Patrinely, J. R., Stone, C. A., Jr, Zimmerman, E., Donald, R. R., Chang, S. S., Berkowitz, S. T., Finn, A. P., Jahangir, E., Scoville, E. A., Reese, T. S., Friedman, D. L., Bastarache, J. A., van der Heijden, Y. F., Wright, J. J., Ye, F., Carter, N., Alexander, M. R., Choe, J. H., ... Johnson, D. B. (2023). Accuracy and Reliability of Chatbot Responses to Physician Questions. *JAMA Network Open*, 6(10), e2336483. <https://doi.org/10.1001/jamanetworkopen.2023.36483>
- Haugeland, I. K. F., Følstad, A., Taylor, C., & Bjørkli, C. A. (2022). Understanding the user experience of customer service chatbots: An experimental study of chatbot interaction design. *International Journal of Human-Computer Studies*, 161, 102788. <https://doi.org/10.1016/j.ijhcs.2022.102788>
- Khenrouche, F., Elmir, Y., Djebbari, N., Himeur, Y., & Amira, A. (2023). *Revolutionizing Customer Interactions: Insights and Challenges in Deploying ChatGPT and Generative Chatbots for FAQs* (No. arXiv:2311.09976). arXiv. <https://doi.org/10.48550/arXiv.2311.09976>
- Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W., Rocktäschel, T., Riedel, S., & Kiela, D. (2021). *Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks* (No. arXiv:2005.11401). arXiv. <https://doi.org/10.48550/arXiv.2005.11401>
- Mageira, K., Pittou, D., Papasalouros, A., Kotis, K., Zangogianni, P., & Daradoumis, A. (2022). Educational AI Chatbots for Content and Language Integrated Learning. *Applied Sciences*, 12(7), 3239. <https://doi.org/10.3390/app12073239>
- Nichifor, E., Trifan, A., & Nechifor, E. M. (2021). Artificial Intelligence in Electronic Commerce: Basic Chatbots and Consumer Journey. *The AMFITEATRU ECONOMIC Journal*, 23(56), 1-87.
- Packer, C., Wooders, S., Lin, K., Fang, V., Patil, S. G., Stoica, I., & Gonzalez, J. E. (2024). *MemGPT: Towards LLMs as Operating Systems* (No. arXiv:2310.08560). arXiv. <https://doi.org/10.48550/arXiv.2310.08560>
- Peña-Torres, J. A., Giraldo-Alegría, S., Arango-Pastrana, C. A., & Bucheli, V. A. (2022). Un chatbot para asistir a las necesidades de información en tiempos de COVID-19. *Ingeniería y Competitividad*, 24(1), Article 1. <https://doi.org/10.25100/iyc.v24i1.11004>
- Peyton, K., & Unnikrishnan, S. (2023). A comparison of chatbot platforms with the state-of-the-art sentence BERT for answering online student FAQs. *Results in Engineering*, 17, 100856. <https://doi.org/10.1016/j.rineng.2022.100856>
- Ray, P. P. (2023). ChatGPT: A comprehensive review on background, applications, key challenges, bias, ethics, limitations and future scope. *Internet of Things and Cyber-Physical Systems*, 3, 121-154. <https://doi.org/10.1016/j.iotcps.2023.04.003>
- Selamat, M. A., & Windasari, N. A. (2021). Chatbot for SMEs: Integrating customer and business owner perspectives. *Technology in Society*, 66, 101685. <https://doi.org/10.1016/j.techsoc.2021.101685>

- Shahsavar, Y., & Choudhury, A. (2023). User Intentions to Use ChatGPT for Self-Diagnosis and Health-Related Purposes: Cross-sectional Survey Study. *JMIR Human Factors*, 10(1), e47564. <https://doi.org/10.2196/47564>
- Sulaiman, R. B. (2022). *AI Based Chatbot: An Approach of Utilizing On Customer Service Assistance* (No. arXiv:2207.10573). arXiv. <https://doi.org/10.48550/arXiv.2207.10573>
- Suresh, S. K., & Shunmugapriya, P. (2024). *Towards smaller, faster decoder-only transformers: Architectural variants and their implications* (No. arXiv:2404.14462). arXiv. <https://doi.org/10.48550/arXiv.2404.14462>
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). *Attention Is All You Need* (No. arXiv:1706.03762). arXiv. <https://doi.org/10.48550/arXiv.1706.03762>