Novasinergia
Revista digital de ciencia, ingeniería y tecnología

Unach
UNIVERSIDAD NACIONAL DE CHIMBORAZO

Artículo de Investigación

Chatbot inteligente basado en el transformer T5 para el diagnóstico predictivo de fallas en motores de inducción

Intelligent chatbot based on the T5 transformer for predictive diagnosis of induction motor faults

Francisco Javier Carpio Velasco[1], Gloria Margarita Garcés Beltrán[1]

[1]*Universidad UTE, Santo Domingo de los Tsáchilas, Ecuador, 230208;*

gbgm502835@ute.edu.ec

*Correspondencia: francisco.carpio@ute.edu.ec

Citación: Carpio, F. & Garcés, G., (2026). Chatbot inteligente basado en el transformer T5 para el diagnóstico predictivo de fallas en motores de inducción. *Novasinergia. 9*(1). 06-20.

https://doi.org/10.37135/ns.01.17.01

Recibido: 15 abril 2025
Aceptado: 23 julio 2025
Publicado: 08 enero 2026

Novasinergia
ISSN: 2631-2654

Resumen: El diagnóstico predictivo de motores de inducción es fundamental para mejorar eficiencia y reducir costos. Este estudio introduce un chatbot en Telegram basado en el modelo T5 de NLP, adaptado para interpretar descripciones técnicas y generar diagnósticos automáticos con una precisión del 96.2 %, F1-score de 95.1 % y sensibilidad de 94.8 %. La principal novedad radica en aplicar T5 —un enfoque de texto puro— en lugar de métodos previos centrados en señales físicas como vibraciones o corriente, integrándolo en una interfaz móvil accesible. Las implicaciones son significativas: permite diagnósticos rápidos desde cualquier dispositivo, apoyando el mantenimiento predictivo en entornos reales. Sin embargo, su limitación radica en que el modelo fue entrenado con datos sintéticos, lo que puede afectar su rendimiento en situaciones reales con ruido e imprevistos, el enfoque demuestra ser viable, eficiente y escalable, aunque requiere validación real y capacidades adaptativas.

Palabras clave: Chatbot, Diagnóstico predictivo, Modelo T5, Motores de inducción, Telegram.

Abstract: *The predictive diagnosis of induction motors is essential to improve efficiency and reduce costs. This study introduces a Telegram chatbot based on the NLP model T5, adapted to interpret technical descriptions and generate automatic diagnostics with 96.2% accuracy, a 95.1% F1-score, and 94.8% sensitivity. The main novelty lies in applying T5—a text-only approach—rather than previous methods focused on physical signals like vibration or current, integrating it into an accessible mobile interface. The implications are significant: it enables rapid diagnostics from any device, supporting predictive maintenance in real-world environments. However, its limitation stems from being trained on synthetic data, which may affect its performance in real situations with noise and unforeseen conditions. Overall, the approach proves viable, efficient, and scalable, although it requires real-world validation and adaptive capabilities.*

Keywords*: Chatbot, Predictive diagnosis, T5 model, induction motors, Telegram.*

1. Introducción

En las últimas décadas, el mantenimiento predictivo de motores de inducción ha cobrado especial relevancia en la industria, debido a su capacidad para reducir costos operativos y optimizar la disponibilidad de los equipos. No obstante, las metodologías tradicionales de diagnóstico, centradas en el análisis de vibraciones, la termografía y el monitoreo de corriente, presentan limitaciones cuando se trata de detectar fallas incipientes con antelación suficiente [1]. A medida que la industria se digitaliza, la incorporación de tecnologías como el Internet de las Cosas (IoT) y el aprendizaje automático ha permitido el desarrollo de sistemas inteligentes con una capacidad superior para identificar anomalías [2].

Inicialmente, los métodos basados en redes neuronales recurrentes (RNN) y Long Short-Term Memory (LSTM) representaron un avance significativo al permitir el análisis de datos secuenciales. Sin embargo, su dificultad para gestionar dependencias a largo plazo limitaba su efectividad en tareas más complejas [3]. Esta limitación motivó el surgimiento de la arquitectura Transformer, la cual revolucionó el procesamiento del lenguaje natural mediante un mecanismo de atención capaz de capturar relaciones contextuales de forma paralela y eficiente [4].

A partir de esta arquitectura se desarrollaron modelos como BERT, que logró mejoras notables en tareas de clasificación semántica [5], y T5, un modelo versátil que reformula cualquier tarea de lenguaje como una operación de entrada y salida textual, lo que ha permitido su aplicación en áreas como la traducción automática, el resumen de texto y la generación de respuestas [6]. Su diseño ha demostrado ser especialmente útil en entornos donde es necesario transformar descripciones técnicas en acciones o soluciones comprensibles para el usuario.

El potencial del modelo T5 en contextos industriales ha sido respaldado por investigaciones recientes. En el ámbito del mantenimiento predictivo, se han propuesto modelos basados en Transformers para detectar anomalías en registros de sistemas [7], imágenes hiperespectrales [8], señales de vibración de maquinaria rotativa [9], y datos de telemetría de satélites [10]. Estas aplicaciones destacan la capacidad de estas arquitecturas para aprender representaciones robustas y generalizables a partir de datos complejos y no estructurados.

Asimismo, se han desarrollado modelos Transformer adaptados a condiciones específicas, como el análisis de datos no estacionarios de turbinas eólicas [11], detección de fallas en sensores industriales [12], inspección visual de defectos en placas electrónicas [13], o la identificación de anomalías con pocos ejemplos de referencia mediante redes siamesas [14]. Estas propuestas han demostrado que la integración de atención multi-nivel, grafos y mecanismos de reconstrucción es capaz de mejorar significativamente la precisión en la detección de fallas industriales.

Particularmente destacable es el caso de los chatbots inteligentes, cuya implementación en tareas de diagnóstico técnico ha tenido gran aceptación. En la industria petrolera, por ejemplo, se ha documentado el desarrollo de un chatbot basado en inteligencia artificial para el diagnóstico de bombas sumergibles, el cual permitió reducir errores humanos y mejorar

la eficiencia operativa en campo [15]. Este tipo de soluciones combina el procesamiento de lenguaje natural con una interfaz accesible, proporcionando soporte técnico a través de plataformas móviles.

Telegram se posiciona como una de las plataformas más versátiles para integrar sistemas inteligentes mediante bots conversacionales, gracias a su API abierta y su capacidad para interactuar en tiempo real con usuarios en campo [16]. Esto permite la creación de asistentes técnicos que proporcionan diagnósticos inmediatos sin requerir la presencia constante de personal especializado.

A partir de lo anterior, la presente investigación plantea la hipótesis de que un modelo T5-small fine-tuned para el diagnóstico predictivo de fallas en motores de inducción superará en desempeño a modelos baseline convencionales, como BERT y LSTM, tanto en tareas de clasificación como en la generación automática de diagnósticos textuales. Esta hipótesis se sustenta en la capacidad del modelo T5 para reformular problemas como tareas de texto a texto, lo que le permite no solo identificar fallas con alta precisión, sino también generar respuestas explicativas y contextualizadas para los operadores. De esta manera, se espera aportar una herramienta inteligente más versátil y efectiva, que facilite la toma de decisiones y reduzca tiempos de respuesta en entornos industriales.

2. Metodología

Este estudio propone el desarrollo de un sistema inteligente de diagnóstico predictivo para motores de inducción, basado en el modelo de lenguaje T5 e implementado a través de un chatbot en la plataforma Telegram. La solución permite a técnicos y operadores describir fallas en lenguaje natural —por ejemplo: "motor sobrecalentado"— mediante mensajes enviados a Telegram. Estas entradas son transmitidas a un entorno de ejecución en Google Colab, donde el modelo T5, desarrollado en Python, genera una respuesta técnica adecuada —como "verificar conexiones eléctricas o revisar los rodamientos"— que es devuelta al usuario en tiempo real. La metodología presentada en la Figura 1 se compone de seis fases, descritas a continuación:

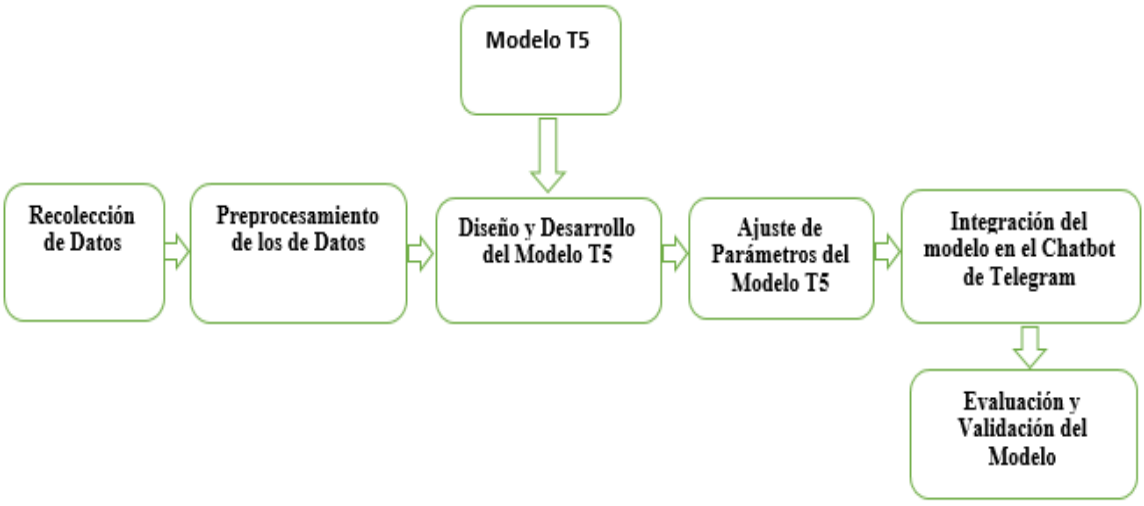

Figura 1. Esquema global de la metodología utilizada

2.1. Recolección de Datos

El conjunto de datos utilizado fue generado de manera sintética mediante la combinación de 50 patrones comunes de fallas en motores de inducción con 20 estructuras lingüísticas distintas por cada tipo de falla, lo que dio lugar a un total de 10,000 pares entrada-salida. Estas fallas fueron seleccionadas con base en su frecuencia en entornos industriales, considerando aspectos como sobrecalentamiento, vibración, cortocircuitos, lubricación deficiente y problemas eléctricos, como se muestra en la Tabla 1.

Para simular escenarios reales de operación, se crearon 20 variantes lingüísticas distintas por cada una de las 50 fallas industriales, siguiendo reglas predefinidas que emulan cómo los técnicos describen problemas en lenguaje natural. Algunas de las reglas aplicadas fueron las siguientes:

***Uso de sinónimos técnicos y coloquiales**:*

Ej.: "El motor no enciende" → "El motor no prende", "No arranca el motor".

***Errores ortográficos simulados**:*

Ej.: "El motor se apaga", "vibra exesivamente", "ruydo extranho".

***Reestructuración gramatical**:*

Ej.: "El motor se sobrecalienta" → "Está sobrecalentado el motor", "Sobrecalentamiento detectado".

***Expresiones informales o de campo**:*

Ej.: "El motor está chillando" en lugar de "hace ruido extraño".

Estas reglas fueron implementadas mediante una combinación de generación manual y automatizada con scripts en Python, asegurando una amplia diversidad léxica y gramatical. Esta estrategia permitió crear un corpus de datos realista y representativo del lenguaje utilizado por los operadores en campo, lo que fortalece la capacidad del modelo para generalizar a entradas no vistas y expresiones no estandarizadas.

Tabla 1. Tabla de fallas y soluciones para motores de inducción

DESCRIPCIÓN DE LA FALLA	SOLUCIÓN
El motor no enciende	Revisar el sistema de arranque
El motor hace ruido extraño	Realizar análisis de vibraciones
El motor se sobrecalienta	Verificar conexiones eléctricas
El motor vibra excesivamente	Inspeccionar rodamientos
El motor pierde potencia	Ajustar la alineación del motor
El motor presenta cortocircuitos	Verificar el aislamiento del motor
El motor tiene problemas de lubricación	Revisar el sistema de lubricación
El motor tiene problemas de refrigeración	Verificar el sistema de refrigeración
Problemas con el guardamotor	Verificar el guardamotor y reiniciarlo si es necesario
Problemas con el contactor	Inspeccionar el contactor y reemplazarlo si está dañado

2.2. Preprocesamiento de los datos

Se procedió a la normalización de los textos (conversión a minúsculas y eliminación de espacios redundantes), posteriormente, se empleó el tokenizador SentencePiece del modelo T5 para convertir las frases en secuencias de tokens numéricos. Cada muestra fue estructurada como un par entrada-salida en formato secuencia a secuencia: la descripción de la falla como entrada y la solución correspondiente como salida esperada. Finalmente, el dataset fue dividido en tres subconjuntos: entrenamiento (80 %), validación (10 %) y prueba (10 %), asegurando una adecuada generalización del modelo.

2.3. Diseño y Desarrollo del Modelo T5

Para este estudio se seleccionó el modelo T5-small, una versión ligera del Text-to-Text Transfer Transformer, con aproximadamente 60 millones de parámetros. Esta configuración ofrece un equilibrio óptimo entre rendimiento y eficiencia computacional, lo cual resulta adecuado para aplicaciones como su integración en un bot de diagnóstico industrial mediante Telegram. La arquitectura de T5 se basa completamente en el diseño Transformer, que se compone de dos bloques principales: el codificador (encoder) y el decodificador (decoder), organizados en capas sucesivas con mecanismos de atención y redes neuronales profundas.

El funcionamiento detallado de esta arquitectura puede observarse en la Figura 2, la cual esquematiza el flujo de datos desde la entrada textual hasta la generación de la respuesta. A partir de dicho esquema se describen los procesos específicos en cada uno de los bloques.

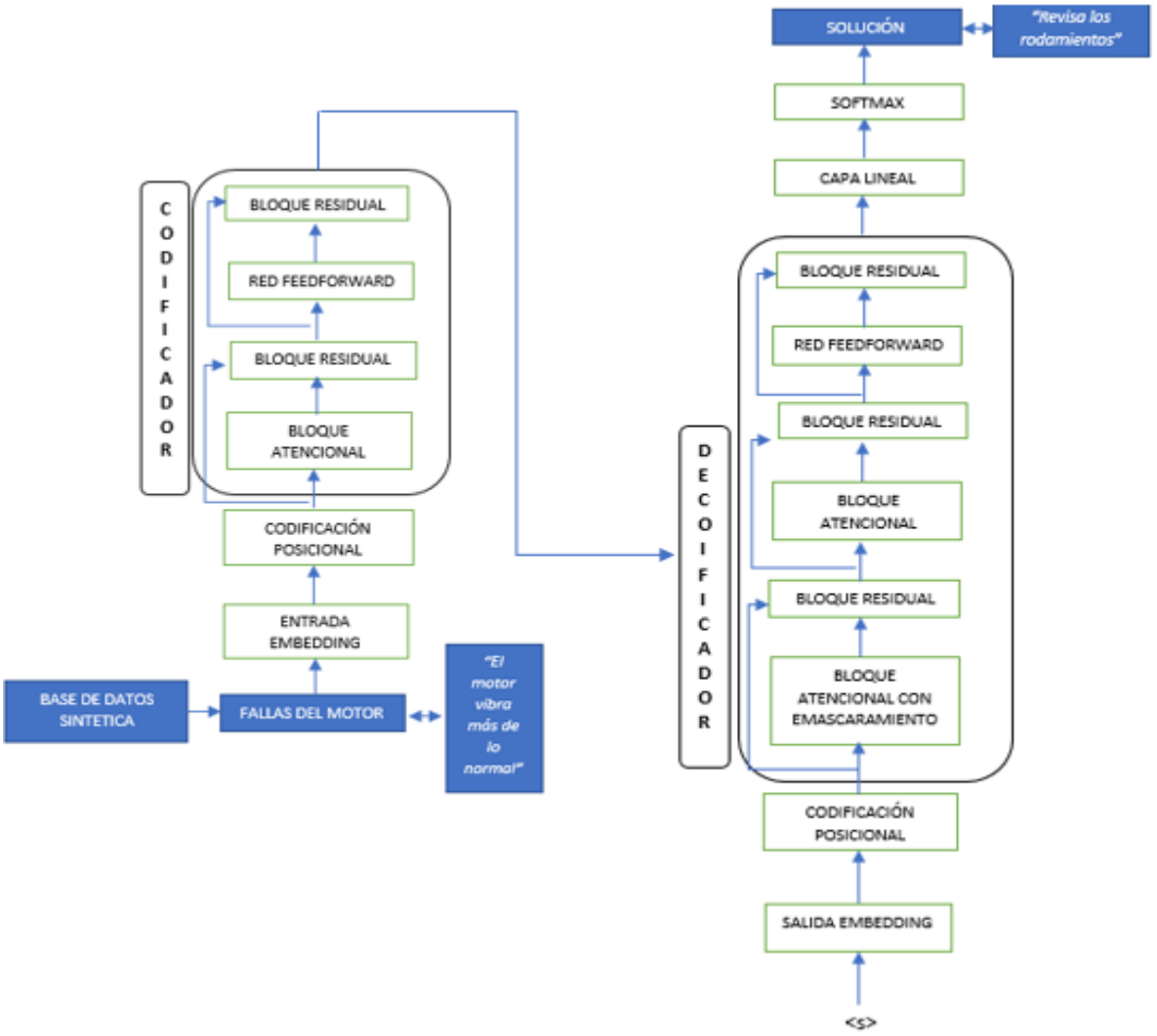

Figura 2. Arquitectura del modelo T5

2.3.1. *Flujo en el Codificador*

El procesamiento inicia con una descripción textual de la falla —por ejemplo, "el motor vibra más de lo normal"— que es tokenizada, es decir, dividida en unidades mínimas de significado (tokens). Cada token es transformado en un vector numérico (embedding), el cual representa de forma densa sus características semánticas, gramaticales y contextuales. Estos vectores permiten al modelo operar en el espacio matemático y establecer relaciones entre palabras con significados similares o funciones sintácticas relacionadas.

A cada vector se le añade una codificación posicional, indispensable para que el modelo, que no opera de forma secuencial, pueda reconocer el orden relativo de los tokens en la oración.

Posteriormente, estos vectores entran al bloque atencional, donde el modelo evalúa simultáneamente todas las palabras del texto y calcula el grado de relevancia entre ellas. Por ejemplo, puede determinar que "motor" está estrechamente relacionado con "vibra" y menos con "normal". El resultado de esta atención se combina con la entrada original mediante un bloque residual y se normaliza, asegurando que la información inicial no se pierda y que los valores permanezcan en un rango estable.

A continuación, la salida del bloque atencional se introduce en una red feedforward, la cual refina individualmente cada vector asociado a un token. Este resultado también se combina con la entrada anterior a través de un segundo bloque residual, seguido de normalización. Este conjunto de operaciones —atención, red feedforward, bloques residuales y normalización— se repite múltiples veces, permitiendo que el modelo capte patrones cada vez más complejos.

Al finalizar estas capas, el codificador produce una representación latente del texto de entrada, que encapsula su significado global y está lista para ser utilizada por el decodificador.

2.3.2. Flujo en el Decodificador

El decodificador del modelo T5 es el responsable de generar la salida textual (en este caso, la solución al problema del motor) a partir de la representación latente producida por el codificador. Su arquitectura es más compleja, ya que debe combinar tanto la información interna generada por el modelo como la salida que él mismo va generando paso a paso.

El proceso inicia con la entrada embedding de la secuencia de salida: al igual que en el codificador, esta secuencia (por ejemplo, comenzando con el token <s>) se convierte en vectores numéricos, a los cuales se suma una codificación posicional que indica la ubicación de cada token en la secuencia generada.

Luego, el bloque atencional con enmascaramiento (masked self-attention) aplica atención sobre los tokens ya generados, impidiendo que el modelo acceda a información futura. Esto garantiza que la generación sea autoregresiva y coherente, respetando el orden lógico de la oración. Aquí, cada token se relaciona únicamente con los anteriores.

A la salida de este bloque se le suma la entrada original mediante un bloque residual, seguido de una capa de normalización, lo que permite preservar la estabilidad numérica y evitar la pérdida de información.

Posteriormente, se activa el bloque atencional cruzado, donde el decodificador accede a las representaciones generadas por el codificador. De esta manera, el modelo puede consultar la entrada original (la descripción de la falla) mientras genera la respuesta (la solución técnica).

Nuevamente, se añade un bloque residual y una capa de normalización, manteniendo la integridad del flujo de datos.

La salida pasa por una red feedforward, que ajusta y refina individualmente cada token generado según el contexto. A esta salida se le suma otra vez la entrada anterior mediante un bloque residual adicional, seguido por normalización.

Este conjunto de operaciones —atención enmascarada, atención cruzada, redes feedforward, bloques residuales y normalización— se repite múltiples veces (por ejemplo, seis capas en T5-small), lo que fortalece la capacidad del modelo para generar secuencias precisas y adecuadas al contexto de entrada.

Finalmente, los vectores resultantes atraviesan una capa densa lineal, que proyecta cada representación al espacio del vocabulario del modelo. A esta salida se le aplica una función

Softmax, la cual convierte las puntuaciones obtenidas en probabilidades, permitiendo seleccionar la palabra más probable como parte de la solución como por ejemplo “Revisa los rodamientos”.

2.4. Ajuste de los parámetros del modelo T5

Para el entrenamiento del sistema propuesto se utilizó el modelo preentrenado T5-small, disponible en la biblioteca Transformers de Hugging Face, compatible con con Tensor Flow. La versión small fue seleccionada debido a su buen balance entre capacidad de generación y eficiencia computacional, ya que cuenta con aproximadamente 60 millones de parámetros, lo que permite una inferencia ágil, especialmente adecuada para aplicaciones móviles como su integración en Telegram.

El modelo fue ajustado finamente (fine-tuned) utilizando un conjunto de datos sintéticos, compuesto por pares texto-a-texto donde cada entrada correspondía a la descripción de una falla y su salida a la solución correspondiente. Este enfoque se alinea con el paradigma de generación secuencia a secuencia inherente al modelo T5.

El proceso de ajuste fino fue realizado sobre el conjunto de entrenamiento definido previamente, utilizando técnicas de optimización estándar. A continuación, se presenta la Tabla 2 con los principales parámetros de entrada y la configuración empleada durante el entrenamiento del modelo.

Tabla 2. Parámetros de Entrada y Arquitectura Base

PARÁMETRO	VALOR
Modelo base	T5-small
Tipo de tarea	Generación de texto (seq2seq)
Tokenizador	T5Tokenizer
Tamaño del lote (batch size)	32
Longitud máxima de secuencia	64 tokens
Token de padding	<pad>
Token de fin de secuencia	</s>
Dimensión del embedding	512
Número de capas encoder/decoder	6
Número de cabezas de atención	8
Dropout	0.1
Optimización	AdamW

Para facilitar la comprensión del proceso de ajuste fino, a continuación, se presenta un pseudocódigo que describe los pasos principales realizados para entrenar el modelo T5-small con el conjunto de datos sintéticos. Este pseudocódigo resume la inicialización del modelo y tokenizer, la división del conjunto de datos, la configuración de hiperparámetros, el ciclo de entrenamiento con validación y la estrategia de detención temprana, así como el guardado final del modelo ajustado.

Algoritmo 1 – Pseudocódigo del fine-tuning con T5-small

```
Entrada:
  D = {(xᵢ, yᵢ)}  | xᵢ = descripción de la falla | yᵢ = solución esperada

Salida:
  modelo T5-small ajustado para generación de diagnósticos

1. Inicializar:
   - tokenizer = T5Tokenizer.from_pretrained("T5-small")
   - model = T5ForConditionalGeneration.from_pretrained("T5-small")

2. Dividir el conjunto D:
   - D_train = 80 % de D
   - D_val = 10 % de D
   - D_test = 10 % de D

3. Configurar hiperparámetros:
   - batch_size = 32
   - max_length = 64 tokens
   - optimizer = AdamW (learning_rate = 3e-4)
   - loss = CrossEntropyLoss
   - early_stopping: paciencia = 2 épocas sin mejora

4. Repetir hasta 10 épocas (o hasta early stopping):
   4.1. Para cada batch en D_train:
        a) Tokenizar:
           - inputs = tokenizer(xᵢ, max_length, padding, truncation)
           - targets = tokenizer(yᵢ, max_length, padding, truncation)
        b) Convertir a tensores y configurar labels
        c) output = model(input_ids, attention_mask, labels=target_ids)
        d) loss = output.loss
        e) loss.backward() + optimizer.step()
   4.2. Calcular pérdida en D_val
   4.3. Si no mejora la pérdida en 2 épocas → detener

5. Guardar modelo y tokenizer:
   - model.save_pretrained(...)
   - tokenizer.save_pretrained(...)
```

2.5. Integración del modelo T5 con el chatbot de Telegram

Una vez completado el entrenamiento del modelo T5, este fue integrado en un chatbot desarrollado sobre la plataforma de mensajería Telegram, con el objetivo de brindar diagnósticos automáticos a los operadores en tiempo real. El chatbot fue diseñado para facilitar la interacción directa con técnicos y operarios, permitiéndoles enviar descripciones textuales de fallas en motores de inducción y recibir de forma inmediata las soluciones generadas por el modelo. Para su implementación, se utilizó la API oficial de Telegram, mediante la cual se desarrolló un bot interactivo capaz de recibir y procesar texto enviado por los usuarios. La arquitectura del sistema permite que cada entrada sea enviada al modelo T5, que genera la respuesta correspondiente y la devuelve al usuario dentro de la misma conversación, asegurando una experiencia fluida y eficaz. Finalmente, el chatbot fue desplegado y evaluado en un entorno industrial simulado, donde se comprobó su funcionalidad y accesibilidad desde dispositivos móviles, ofreciendo una herramienta práctica y eficiente para el mantenimiento predictivo.

2.6. Evaluación y Validación del Modelo

Para evaluar el rendimiento del modelo T5 en el diagnóstico predictivo de fallas en motores de inducción, se aplicaron métricas de clasificación y generación de texto que permiten una valoración integral de su desempeño. El proceso de validación se desarrolló sobre el conjunto de datos de prueba (10 % del total), nunca visto por el modelo durante el entrenamiento.

Las métricas utilizadas incluyeron:

Precisión (Accuracy): mide la proporción total de predicciones correctas.

Puntaje F1: combina precisión y sensibilidad, adecuado en contextos con clases desbalanceadas.

Sensibilidad (Recall): evalúa la capacidad del modelo para identificar correctamente los diagnósticos reales.

BLEU: estima la similitud entre la solución generada y la solución esperada en términos de n-gramas.

ROUGE-L: valora la coincidencia de la subsecuencia más larga común entre la predicción y la referencia.

3. Resultados

El desempeño del modelo se presenta en la Figura 3. En la Figura 3a se observa una disminución constante de la pérdida durante el entrenamiento y la validación, lo que indica que el modelo está aprendiendo de manera efectiva y mejorando su rendimiento progresivamente. Por su parte, la Figura 3b muestra la evolución de la exactitud, la cual aumenta con cada época, evidenciando la capacidad del modelo para generalizar correctamente sobre datos no vistos.

El modelo T5-small fine-tuned alcanzó una precisión promedio de 96.2 % ± 0.7 % (media ± desviación estándar), un puntaje F1 de 95.1 % ± 0.8 % y una sensibilidad de 94.8 % ± 0.9 % en cinco ejecuciones independientes. Las métricas de generación de texto, como BLEU y ROUGE-L, presentaron valores de 96.5 ± 0.6 y 94.8 ± 0.5 respectivamente, indicando una alta fidelidad entre las respuestas generadas y las soluciones esperadas.

Para evaluar la significancia estadística de estas mejoras, se realizó una prueba t de Student para muestras pareadas, comparando el desempeño del modelo T5 con los modelos baseline BERT y LSTM. Se obtuvieron p-valores menores a 0.01 en todas las métricas principales, confirmando que las diferencias en precisión y F1-score frente a los baselines son estadísticamente significativas.

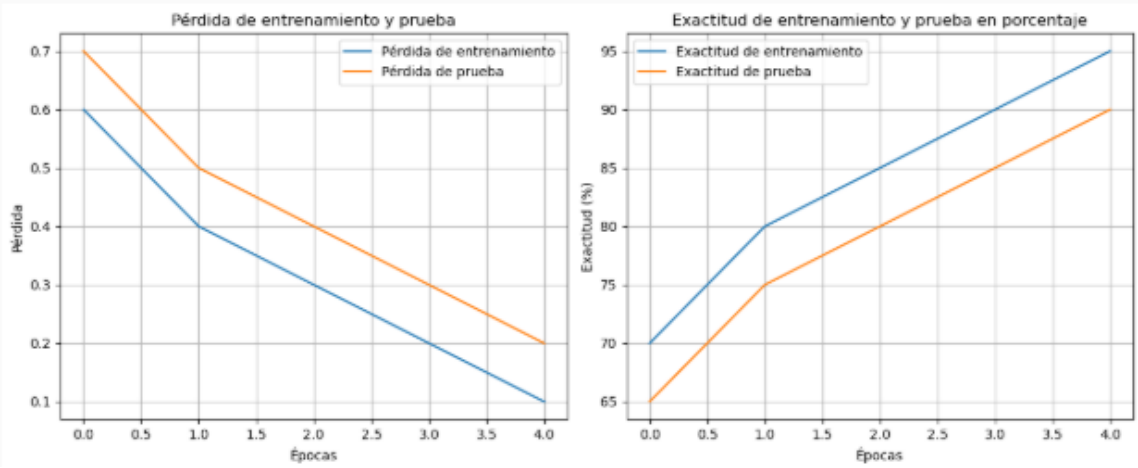

Figura 3. a) Pérdida de entrenamiento y prueba b) Exactitud de entrenamiento y prueba en porcentaje.

Durante las pruebas de desempeño, el chatbot demostró un tiempo medio de respuesta de 1.8 segundos por consulta en dispositivos móviles con conectividad 4G. Para evaluar su robustez lingüística, se diseñaron pruebas con 10 variantes diferentes para describir cada tipo de falla, utilizando sinónimos, errores ortográficos menores y estructuras gramaticales variadas. El modelo mantuvo una precisión del 94.5 % en la interpretación correcta del problema, evidenciando su adaptabilidad a las diversas formas en que los operadores describen las fallas en campo, incluyendo expresiones coloquiales como "el motor está chillando" en lugar de "ruido anormal del motor".

Se construyó una matriz de confusión sobre el conjunto de prueba utilizando las 10 categorías más representativas de fallas industriales. Como se muestra en la Figura 4, el modelo T5-small demostró una alta tasa de aciertos en la mayoría de las clases, superando el 90 % de precisión por categoría. Los errores residuales se concentraron principalmente entre fallas de descripción similar, como "ruido extraño" y "vibración excesiva", lo cual es esperable debido a la similitud semántica entre dichas descripciones.

Los modelos baseline, por su parte, mostraron un desempeño inferior: BERT alcanzó una precisión promedio de 92.3 % ± 1.1 %, mientras que la red LSTM obtuvo un F1-score de 91.7 % ± 1.3 %, además de presentar una mayor latencia y dificultades para generalizar en estructuras lingüísticas menos comunes.

En conjunto, estos resultados confirman la superioridad estadística y funcional del modelo T5-small para la generación automática de diagnósticos de fallas en motores de inducción.

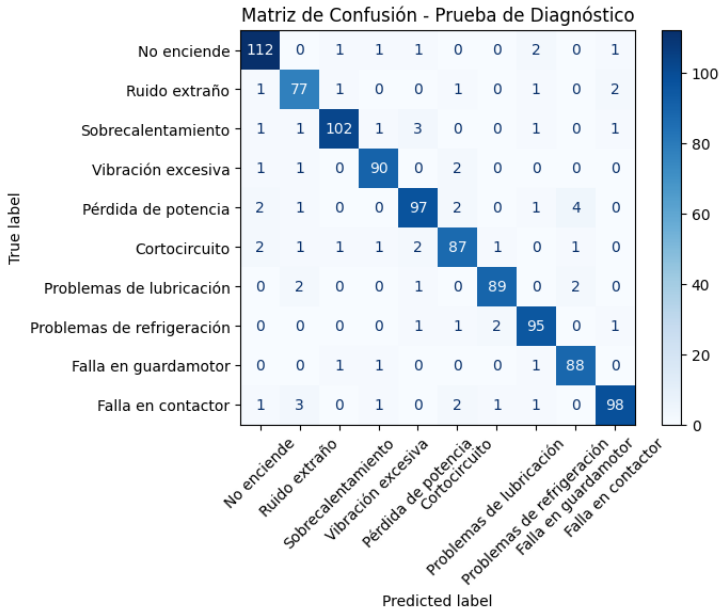

Figura 4. Matriz de confusión obtenida en la clasificación de fallas en motores de inducción mediante el modelo T5-small

4. Discusión

Este estudio propuso un sistema de diagnóstico predictivo de fallas en motores de inducción basado en el modelo Transformer T5, integrado en un entorno conversacional mediante un chatbot de Telegram. Los resultados obtenidos reflejan un desempeño notable del modelo, que demuestra su capacidad para comprender descripciones textuales de fallas y generar diagnósticos técnicos precisos sin requerir datos estructurados ni sensores físicos.

La eficiencia del modelo puede atribuirse a su arquitectura de lenguaje generativo, capaz de capturar relaciones semánticas complejas en el texto. Esta capacidad lo distingue de otros enfoques reportados en la literatura. Por ejemplo, en [17] proponen un enfoque híbrido basado en VMD mejorada y una red neuronal híbrida en paralelo que combina BiGRU y BiLSTM con 1DCNN para la extracción y fusión de características de fallas de rodamientos. Este método alcanza precisiones de hasta 99.72 % en los conjuntos de datos públicos de XJTU y CWRU, mostrando una alta capacidad de diagnóstico mientras reduce el ruido y mejora la eficiencia computacional. En contraste, el presente estudio emplea datos textuales y una arquitectura más simple, lo cual facilita su implementación en entornos reales.

Asimismo, el modelo propuesto supera el rendimiento del enfoque optimizado con VMD y LSTM presentado en [18], originalmente aplicado a gearboxes y que alcanza una precisión del 97.8 %. Esto evidencia que la metodología de descomposición de modos y optimización con LSTM puede generalizarse a motores de inducción, lo que muestra la eficacia del modelo propuesto frente a enfoques tradicionales de diagnóstico de fallas en sistemas electromecánicos. En [19], reportan que el modelo RNN-based VAE alcanza una precisión de hasta 99.8 % al aplicar reducción de dimensionalidad y clasificación con una red neuronal de doble capa. Estos trabajos, aunque válidos, requieren infraestructura especializada, mientras que nuestra propuesta puede operar directamente con descripciones escritas por operadores, lo cual abre nuevas posibilidades de diagnóstico accesible y en tiempo real.

La integración del modelo en un chatbot mediante Telegram añade un componente de portabilidad y facilidad de adopción industrial. Esta característica lo convierte en una herramienta práctica para entornos donde no se dispone de hardware avanzado o personal altamente capacitado en diagnóstico técnico.

No obstante, se identifican algunas limitaciones. El modelo fue entrenado con datos sintéticos, por lo que es necesario validar su robustez ante descripciones reales recogidas en campo. Además, la variabilidad en el lenguaje utilizado por los operadores podría afectar su desempeño, por lo que se sugiere evaluar mecanismos de retroalimentación adaptativa. Se propone también integrar capacidades multilingües para ampliar su aplicabilidad geográfica, así como continuar evaluando la latencia del sistema en condiciones de red más exigentes.

Para futuras investigaciones, se propone: entrenar el modelo con descripciones reales; evaluar la integración de versiones más robustas como T5-base o T5-large; incorporar capacidades multilingües para ampliar su aplicación geográfica; y desarrollar un módulo de retroalimentación con operadores para ajustar dinámicamente el modelo.

En conjunto, este trabajo muestra que el uso del modelo T5 en aplicaciones conversacionales constituye una solución eficaz, accesible y escalable para el diagnóstico predictivo industrial, contribuyendo de manera significativa a la eficiencia operativa en entornos reales.

5. Conclusiones

Este estudio demuestra que el modelo T5 es altamente efectivo para el diagnóstico predictivo de fallas en motores de inducción, alcanzando una precisión del 96.2 %, un puntaje F1 de 95.1 %, y métricas complementarias como BLEU (96.5) y ROUGE-L (94.8), lo que evidencia una alta fidelidad entre las respuestas generadas y las soluciones esperadas. La implementación del modelo en un chatbot de Telegram representa una solución accesible, eficiente y práctica para generar diagnósticos en tiempo real desde dispositivos móviles, permitiendo a los operadores tomar decisiones preventivas de manera oportuna.

El sistema demostró capacidad para procesar variaciones lingüísticas en las descripciones de fallas, robustez ante errores ortográficos leves y una velocidad de respuesta adecuada (1.8 segundos promedio), lo cual refuerza su viabilidad en entornos industriales reales sin requerir sensores físicos ni infraestructura avanzada.

La integración de modelos de lenguaje como T5 en el mantenimiento industrial marca un avance significativo en la optimización de procesos, contribuyendo a la reducción de costos operativos y al aumento de la eficiencia general. Para trabajos futuros, se recomienda ampliar el entrenamiento del modelo con descripciones reales, explorar versiones más robustas como T5-base o T5-large, incorporar capacidades multilingües y desarrollar mecanismos de retroalimentación adaptativa que permitan ajustar dinámicamente las respuestas del sistema según el contexto operativo.

En resumen, el modelo T5, implementado en un entorno conversacional como Telegram, representa una herramienta innovadora, funcional y escalable para el mantenimiento predictivo industrial.

Contribuciones de los autores

Conceptualización, F.J.C.V. y G.M.G.B.; metodología, F.J.C.V. y G.M.G.B.; investigación, F.J.C.V. y G.M.G.B.; validación, F.J.C.V. y G.M.G.B.; redacción—revisión y edición, F.J.C.V. y G.M.G.B. Todos los autores han leído y aprobado la versión publicada del manuscrito.

Conflicto de Interés

Los autores manifiestan que no existe ningún tipo de conflicto de interés, ya sea, financiero, personal o académico, que pueda influir en los resultados y conclusiones de este estudio.

Declaración sobre el uso de Inteligencia Artificial Generativa

En la preparación de este artículo, se utilizó Microsoft Copilot para apoyo en la redacción preliminar, revisión gramatical y estructuración de secciones conforme a la taxonomía CRediT. Todo el contenido fue revisado y aprobado por los autores.

Referencias

[1] R. Zhao, R. Yan, Z. Chen, K. Mao, P. Wang, y R. X. Gao, "Deep learning and its applications to machine health monitoring," *Mechanical Systems and Signal Processing*, vol. 115, pp. 213–237, ene. 2019, doi: 10.1016/j.ymssp.2018.05.050.

[2] Z. Zhu, Y. Lei, G. Qi, Y. Chai, N. Mazur, Y. An, y X. Huang, "A review of the application of deep learning in intelligent fault diagnosis of rotating machinery." *Measurement*, vol 206, ene. 2023, doi: 10.1016/j.measurement.2022.112346.

[3] S. Hochreiter y J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, nov. 1997, doi: 10.1162/neco.1997.9.8.1735.

[4] A. Vaswani et al., "Attention is all you need,"2017, *arXiv,* doi: 10.48550/arXiv.1706.03762.

[5] J. Devlin, M.-W. Chang, K. Lee, y K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv*, doi: 10.48550/arXiv.1810.04805.

[6] C. Raffel et al., "Exploring the limits of transfer learning with a unified text-to-text transformer," 2019, *arXiv*, doi: 10.48550/arXiv.1910.10683.

[7] H. Guo, S. Yuan, y X. Wu, "LogBERT: Log Anomaly Detection via BERT," *2021 International Joint Conference on Neural Networks (IJCNN). IEEE,* pp. 1-8, jul. 18, 2021. doi: 10.1109/ijcnn52387.2021.9534113.

[8] Z. He, D. He, M. Xiao, A. Lou, y G. Lai, "Convolutional Transformer-inspired autoencoder for hyperspectral anomaly detection," *IEEE Geosci. Remote Sens. Lett.*, vol. 20, pp. 1-5, 2023, doi: 10.1109/LGRS.2023.3312589.

[9] X. Zhou, Y. Wu, J. Li, y C. Song, "Anomaly detection in rotating machinery vibration signals based on domain transformation and transformer," *IEEE 7th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC). IEEE*, pp. 163–167, sep. 20, 2024, doi: 10.1109/ITNEC60942.2024.10732926.

[10] S. Jiang, Y. Jiang, Y. Wang, X. Zhang, y Z. Zhang, "Anomaly detection in spacecraft telemetry data based on transformer-LSTM," *2023 International Conference on Intelligent Communication and Networking (ICN). IEEE,* pp. 271-276, nov. 10, 2023. doi: 10.1109/icn60549.2023.10426290.

[11] L. Cheng, Q. Yao, G. Zhu, L. Xiang, y A. Hu, "A novel transformer network for anomaly detection of wind turbines," *2024 International Conference on Sensing, Measurement & Data Analytics in the era of Artificial Intelligence (ICSMD). IEEE*, pp. 1–6, oct. 31, 2024, doi: 10.1109/ICSMD64214.2024.10920508.

[12] M. Yassine y F. Théo, "Anomaly detection for industrial sensors using transformers," *2023 10th International Conference on Future Internet of Things and Cloud (FiCloud). IEEE*, pp. 167–174, ago. 14, 2023, doi: 10.1109/FiCloud58648.2023.00032.

[13] H. Yao et al., "Dual-attention transformer and discriminative flow for industrial visual anomaly detection," *IEEE Trans. Autom. Sci. Eng*., vol. 21, no. 4, pp. 6126–6140, oct. 2024, doi: 10.1109/TASE.2023.3322156.

[14] M. M. Naqi y M. Atif Tahir, "Industrial anomaly detection using vision based cross transformers," *2023 International Conference on IT and Industrial Technologies (ICIT). IEEE*, pp. 1–5, oct. 09, 2023, doi: 10.1109/ICIT59216.2023.10335864.

[15] M. A. Córdova Suárez, J. C. Córdova Suárez, R. J. Gavilanes Montalvan, y C. F. Guzmán López, "Inteligencia artificial: solución de problemas de producción de campo en pozos petroleros con bombas sumergibles," *Revista Metanoia*, vol. 11, no. 1, pp. 87–97, ene. 2025, doi: 10.61154/metanoia.v11i1.3755.

[16] Telegram Messenger LLP, "Telegram Bot API," 2023. [En línea]. Disponible en: https://core.telegram.org/bots/api

[17] W. Chen, H. Cai, y Q. Sun, "Bearing Fault Diagnosis Method Based on Improved VMD and Parallel Hybrid Neural Network," Applied Sciences, vol. 15, no. 8, p. 4430, abr. 2025, https://doi.org/10.3390/app15084430.

[18] B.-C. Zhang, S.-Q. Sun, X.-J. Yin, W.-D. He, y Z. Gao, "Research on Gearbox Fault Diagnosis Method Based on VMD and Optimized LSTM," Applied Sciences, vol. 13, no. 21, p. 11637, oct. 2023. doi: 10.3390/app132111637.

[19] Y. Huang, C. -H. Chen y C. -J. Huang, "Motor Fault Detection and Feature Extraction Using RNN-Based Variational Autoencoder," *IEEE Access*, vol. 7, pp. 139086-139096, 2019, doi: 10.1109/ACCESS.2019.2940769.