|  |  |
| --- | --- |
|  | **Universidad Católica Andrés Bello**  **Facultad de Ingeniería**  **Escuela de Ingeniería Informática** |

**Sistema de Monitoreo Acústico, para Identificar Sonidos y Generar Alertas de Emergencia**

**Trabajo de Grado**

Presentado ante la

**UNIVERSIDAD CATÓLICA ANDRÉS BELLO**

Como parte de los requisitos para optar al título de

**INGENIERO EN INFORMÁTICA**

|  |  |
| --- | --- |
| Realizado por | Naim Arcoiza, Carmelo Jesus  Sotillo Vallejo, César Enrique |
| Tutor | Larez Mata, Jesús José |
| Fecha | Marzo, 2025 |

**Dedicatoria**

Bla bla bla

**Agradecimiento**

Gracias Beyonce por permitirme nacer

**Índice de contenidos**

[Resumen 7](#_Toc190894713)

[Introducción 8](#_Toc190894714)

[Capitulo I. El Problema 10](#_Toc190894715)

[Planteamiento del Problema 10](#_Toc190894716)

[Objetivo General 10](#_Toc190894717)

[Objetivos Específicos 10](#_Toc190894718)

[Alcance 10](#_Toc190894719)

[Limitaciones 11](#_Toc190894720)

[Justificación 12](#_Toc190894721)

[Capitulo II. Marco Metodológico 14](#_Toc190894722)

[Antecedentes de investigación 14](#_Toc190894723)

[Real-time Audio Classification on an Edge Device 14](#_Toc190894724)

[Analysing RMS and peak values of vibration signals for condition monitoring of wind turbine gearboxes 15](#_Toc190894725)

[Metodología para la identificación de eventos sonoros anómalos 16](#_Toc190894726)

[Bases Teóricas 16](#_Toc190894727)

[Dispositivos Edge: 16](#_Toc190894728)

[Raspberry pi 17](#_Toc190894729)

[Python 18](#_Toc190894730)

[Tensorflow 18](#_Toc190894731)

[NestJs 18](#_Toc190894732)

[Inteligencia artificial 19](#_Toc190894733)

[Cadenas de Markov 21](#_Toc190894734)

[Agente 22](#_Toc190894735)

[Bases Legales. 23](#_Toc190894736)

[Capitulo III. Marco Metodológico 25](#_Toc190894737)

[Tipo de Investigación 25](#_Toc190894738)

[Técnicas e Instrumentos de Recolección de Datos 26](#_Toc190894739)

[Metodología de Desarrollo Utilizada 26](#_Toc190894740)

[Referencias Bibliográficas 28](#_Toc190894741)

**Índice de imágenes**

Bla bla bla

|  |  |
| --- | --- |
|  | **Universidad Católica Andrés Bello**  **Facultad de Ingeniería**  **Escuela de Ingeniería Informática** |

**Sistema de Monitoreo Acústico, para Identificar Sonidos y Generar Alertas de Emergencia**

|  |  |
| --- | --- |
| Autor: | Naim Arcoiza, Carmelo Jesus  Sotillo Vallejo, César Enrique |

|  |  |
| --- | --- |
| Tutor: | Larez Mata, Jesús José |
| Fecha: | Marzo, 2025 |

# Resumen

Bla bla bla bla bla y bla

*Palabras clave:* bla, bla, bla. y bla,

**NOTA**

Exposición del problema, objetivo, enfoque teórico, aspectos metodológicos, resultados y conclusiones.

Debe ser: ***preciso***: reflejar el propósito y contenido del informe, sin incluir información que no aparezca en el informe; ***coherente, legible y conciso***

En un solo párrafo, se utiliza interlineado sencillo, SIN sangría. Extensión máxima de **250** palabras

# Introducción

Este trabajo de investigación propone un **sistema distribuido** para la **detección temprana** de emergencias mediante el uso de **inteligencia artificial**, enfocada en proteger no solo a **personas mayores o con discapacidades subyacentes, sino también, a personas que, a raíz de cualquier situación, quedaron en una situación de vulnerabilidad**. El sistema utiliza dispositivos que capturan **sonidos**, clasificándolos y detectando palabras clave relevantes.

Esta información se envía en tiempo real a un servidor central donde se analizan patrones y anomalías con técnicas predictivas mediante **Prophet**. La **importancia** de este proyecto radica en su capacidad de prevenir que se agraven las consecuencias ante un evento que comprometa la salud de un individuo si no se detectan a tiempo, ofreciendo un **monitoreo no invasivo** que respeta la privacidad. Además, beneficia a **familias, cuidadores, instituciones públicas y privadas** y facilita una intervención más rápida de los **servicios de emergencia**, incluso en casos como **violencia doméstica o el monitoreo de personas con depresión**. El **propósito** es crear un entorno más seguro mediante una identificación de situaciones peligrosas y alertar a quienes puedan ofrecer ayuda, contribuyendo a una mejor calidad de vida y una respuesta efectiva ante emergencias.

Para el desarrollo del sistema, se empleó la metodología espiral, que permite un enfoque iterativo y flexible. Cada ciclo del proceso abarca fases de planificación, análisis de riesgos, desarrollo y evaluación, permitiendo ajustes continuos en el diseño y la implementación del sistema. Esta metodología es ideal para proyectos con incertidumbre, ya que facilita la evolución del sistema según la retroalimentación obtenida, asegurando su mejora progresiva a lo largo del desarrollo.

El presente trabajo está estructurado en cinco capítulos, En el Capítulo I, referido al Planteamiento del Problema, se describe la problemática, se establecen los objetivos, justificación, alcance y limitaciones del proyecto. En el Capítulo II, se expone el Marco Teórico donde se recopilan los antecedentes y las bases teóricas que sustentan el trabajo. En el Capítulo III, se presenta el Marco Metodológico, que describe el tipo de investigación, técnicas e instrumentos de recolección de datos, metodología de desarrollo y el procedimiento metodológico. En el Capítulo IV, se expone el Desarrollo y Resultados, donde se describe como el procedimiento metodológico dio respuesta a cada uno de los objetivos planteados, y en el Capítulo V, se presentan las conclusiones y recomendaciones sobre el trabajo realizado. Finalmente se listan las referencias bibliográficas utilizadas, seguidas de los anexos y apéndices.

# Capitulo I. El Problema

## Planteamiento del Problema

### Objetivo General

Desarrollar un sistema para generar alertas en casos de emergencia basado en el monitoreo acústico.

### Objetivos Específicos

1. Analizar los conceptos asociados a la analítica de sonidos ambientales identificando los conceptos necesarios para diseñar el sistema.
2. Diseñar un sistema para generar alertas en casos de emergencia basado en el monitoreo acústico en función del análisis realizado.
3. Implementar el sistema para generar alertas en casos de emergencia basado en el monitoreo acústico según el diseño realizado.
4. Validar el sistema para generar alertas en casos de emergencia basado en el monitoreo acústico con respecto al análisis realizado.
5. Elaborar la documentación técnica y guías del usuario y sistema.

## Alcance

El proyecto tiene como objetivo desarrollar un sistema de monitoreo acústico que perfile el comportamiento de los habitantes de una vivienda, a través del reconocimiento de los sonidos ambientales en las diferentes estancias de la misma. El sistema generará alarmas en tiempo real que alertarán a contactos de emergencia predefinidos, estas notificaciones podrán tener múltiples niveles dependiendo del grado de incertidumbre de la situación. Para ello se llevará a cabo un análisis de los conceptos asociados a la analítica de sonidos donde se identificarán los conceptos necesarios para diseñar el sistema.

Con base en los resultados del análisis, se procederá a diseñar el sistema de monitoreo acústico capaz de reconocer sonidos. El diseño comprende la definición de la arquitectura del sistema, contemplando los componentes esenciales, como la red de micrófonos para la captura de sonidos y los mecanismos de alerta, detección de anomalías y de reconocimiento acústico. El diseño considerará la importancia de la privacidad de los usuarios. Según los resultados del análisis realizado para el desarrollo del sistema de monitoreo acústico, se evaluará si es necesario crear un dataset desde cero, obtener uno de internet y modificarlo según sea necesario, o bien emplear un modelo ya preentrenado.

Una vez completado el diseño, se procederá a la implementación del sistema de monitoreo acústico. Esta fase incluirá la configuración e instalación de los componentes físicos, como los micrófonos distribuidos en las estancias de la vivienda, asegurando una cobertura adecuada para la captura de sonidos relevantes. Los algoritmos de procesamiento de señales y los modelos de inteligencia artificial, definidos en la fase de diseño, serán desarrollados y adaptados para realizar el reconocimiento y clasificación de los sonidos.

Se desarrollarán los mecanismos de alerta y notificación, que emitirán avisos a los contactos de emergencia predefinidos, permitiendo la personalización de los niveles de alarma en función de la detección de anomalías o eventos específicos. Durante esta fase, se asegurará medidas de privacidad para proteger la información sensible capturada por los micrófonos.

Luego se realizará la validación del sistema, se llevarán a cabo pruebas funcionales en ambientes controlados y no controlados para comparar los resultados obtenidos con los objetivos planteados y realizar ajustes en caso de ser necesario.

Finalmente se elaborará la documentación del sistema, manuales de usuario, manuales de sistema, descripción de los componentes, diagramas y cualquier otro documento necesario.

## Limitaciones

En la implementación del sistema, se prevén ciertas limitaciones que podrían influir en el desempeño de su desarrollo. A continuación, se describirán algunas de las restricciones que se podrían presentar:

Limitaciones tecnológicas: La precisión del sistema puede verse afectada por la calidad de los sensores acústicos y la capacidad de procesamiento de los microcontroladores empleados.

Limitaciones de datos: La recolección de suficientes muestras de sonidos para entrenar el modelo puede ser limitada, lo que puede comprometer la eficacia del sistema en escenarios no previstos.

## Justificación

En el contexto actual, donde la seguridad y el bienestar de personas en situaciones vulnerables requieren soluciones innovadoras, surge la necesidad de sistemas de monitoreo no invasivos que equilibren la eficacia tecnológica con el respeto a la privacidad. Este proyecto propone un Sistema de Monitoreo Acústico basado en Inteligencia Artificial como respuesta a estas limitaciones, integrando avances en procesamiento de señales, aprendizaje automático y computación periférica para ofrecer una alternativa segura, escalable y adaptada a realidades locales.

La relevancia de esta investigación radica en su enfoque metodológico y tecnológico. Al emplear la metodología espiral, se garantiza un desarrollo iterativo que incorpora retroalimentación continua, ajustándose a los desafíos propios de proyectos con componentes hardware y software. Además, el uso de modelos de IA ligeros, como TensorFlow Lite, y dispositivos accesibles, como Raspberry Pi, demuestra que es posible implementar soluciones avanzadas con recursos limitados, un aporte significativo en entornos donde la infraestructura tecnológica es escasa. Estos aspectos no solo enriquecen el campo de la ingeniería informática, sino que también establecen un precedente para futuras investigaciones en sistemas embebidos éticos.

Desde una perspectiva social, el sistema beneficia directamente a grupos prioritarios. Por un lado, protege a personas que, por condiciones físicas o emocionales, podrían enfrentar emergencias sin posibilidad de solicitar ayuda inmediata, como casos de caídas, crisis de salud o violencia doméstica. Por otro, facilita la labor de cuidadores, familiares y servicios de emergencia al proporcionar alertas tempranas basadas en análisis acústico en tiempo real, sin comprometer la privacidad mediante el almacenamiento local de dato.

En el ámbito institucional, el proyecto ofrece una herramienta viable para hogares de cuidado, hospitales y comunidades, permitiendo optimizar recursos humanos y económicos. Su arquitectura de bajo costo y código abierto promueve el uso en zonas con acceso limitado a tecnologías avanzadas, contribuyendo a reducir brechas tecnológicas. Asimismo, sienta bases para políticas públicas orientadas a la integración de sistemas automatizados en estrategias de seguridad ciudadana, demostrando que la innovación tecnológica puede coexistir con principios éticos y legales.

Finalmente, esta investigación fortalece el ecosistema académico y productivo nacional. Al desarrollar un prototipo funcional en Venezuela, utilizando hardware disponible localmente y datasets adaptados a contextos reales, se evidencia el potencial de la ingeniería local para generar soluciones originales. Además, la documentación generada servirá como referencia para proyectos futuros, incentivando la investigación en IA aplicada a problemáticas sociales urgentes.

# Capitulo II. Marco Metodológico

Este capítulo presenta las bases teóricas y las investigaciones previas que sustentan el desarrollo del Sistema de Monitoreo Acústico para Identificar Sonidos y Generar Alertas de Emergencia. A través de la revisión de fuentes documentales, se exponen los conceptos y enfoques principales relacionados con el procesamiento de señales acústicas, la clasificación de sonidos mediante técnicas de inteligencia artificial y los sistemas de alertas automatizadas.

El marco teórico no solo proporciona un contexto académico para el proyecto, sino que también justifica las decisiones técnicas y metodológicas adoptadas en el diseño del sistema. En las siguientes secciones se detallan los antecedentes investigativos, los modelos de IA aplicados a la clasificación de sonidos y los fundamentos teóricos que permiten la generación de alertas en situaciones de emergencia.

En las siguientes secciones se detallan los antecedentes y los conceptos clave que sustentan este trabajo.

## Antecedentes de investigación

### Real-time Audio Classification on an Edge Device

Christoffer y David (2021) Abordan la implementación de modelos de aprendizaje automatico en dispositivos edge para la clasificación de audio en tiempo real, Se centra en el uso del modelo YAMNet, que fue reentrenado para detectar eventos acústicos como disparos, rotura de cristales, animales y habla humana. Este modelo puede desplegarse en dispositivos edge en su versión completa con TensorFlow como en versiones optimizadas con TensorFlow Lite, con el objetivo de comparar la precisión, el tiempo de inferencia y el uso de memoria de cada variante​.

Con el propósito de evaluar el desempeño, trabajaron en una serie de experimentos en los que compararon la precisión del modelo en ambas versiones de TensorFlow. Se encontraron con que, aunque existía una pérdida de precisión en la versión lite los resultados eran comparables a los de la versión completa. Esto implica que TensorFlow Lite es una opción viable y crea la posibilidad de trabajar con dispositivos de bajo consumo y recursos limitados como ESP32

Este enfoque no solo demuestra la factibilidad de implementar modelos de clasificación de audio en tiempo real en entornos con recursos restringidos, sino que también abre la puerta a futuras investigaciones que busquen optimizar aún más estos modelos. La capacidad de desplegar soluciones de inteligencia artificial en dispositivos edge amplía las posibilidades de aplicaciones en áreas como la seguridad, la vigilancia y el monitoreo ambiental, permitiendo respuestas más rápidas y reduciendo la dependencia de infraestructuras centralizadas.

### Analysing RMS and peak values of vibration signals for condition monitoring of wind turbine gearboxes

Igba, Alemzadeh, Durugbo & Eiriksson (2016) abordan el monitoreo de condición en aerogeneradores con el objetivo de detectar fallos en las cajas de engranajes mediante el análisis de valores RMS y picos de vibraciones. La investigación propone tres modelos: correlación de señales, vibración extrema e intensidad RMS, validados con datos en dominio del tiempo. A través del uso de la teoría de valores extremos, se identificaron indicadores que permiten detectar fallos en etapas tempranas, lo que facilita la planificación del mantenimiento y minimiza el tiempo de inactividad de los aerogeneradores. Los resultados demostraron que el monitoreo de vibraciones proporciona información clave sobre el estado de los componentes y que la precisión de cada técnica depende de la física de la falla, sugiriendo un enfoque integral que combine distintas estrategias para una evaluación más robusta de la salud de los aerogeneradores.

El uso del valor RMS en este estudio es fundamental para la monitorización de la condición de las cajas de engranajes en aerogeneradores, ya que permite evaluar el nivel global de vibración y detectar fallos progresivos, como el desgaste de rodamientos y grietas en ejes. Igba, Alemzadeh, Durugbo & Eiriksson (2016) destacan que el RMS es una métrica confiable para identificar tendencias anómalas en las vibraciones, lo que facilita la detección temprana de fallos antes de que se conviertan en problemas críticos. A pesar de ciertas limitaciones, como su menor sensibilidad a fallos incipientes en los dientes de los engranajes, el análisis de RMS sigue siendo un pilar clave en la estrategia de mantenimiento basado en condición (CBM), al proporcionar información valiosa sobre la evolución del estado de los componentes mecánicos.

### Metodología para la identificación de eventos sonoros anómalos

Torija, Ruiz y Ramos-Ridao (2008) presentan una metodología para la detección de eventos sonoros anómalos en entornos urbanos. Su trabajo se centra en el análisis de sucesos acústicos que generan incrementos bruscos de energía sonora en el paisaje sonoro urbano, lo que puede provocar molestias significativas en la población expuesta. Según los autores, los eventos sonoros anómalos, como bocinas, gritos, explosiones o disparos, generan una percepción intensificada del ruido ambiental debido a su impacto en la focalización de la atención. Para abordar este problema, desarrollaron un modelo basado en la evaluación del tiempo de estabilización de la medición del ruido ambiental y el análisis del factor cresta, que permite estimar el incremento de energía sonora generado por estos eventos.

El estudio incluyó mediciones en 35 localizaciones de la ciudad de Granada, utilizando un sonómetro Brüel & Kjaer tipo 1. La metodología aplicada permitió definir un evento sonoro anómalo como aquel que provoca un incremento del nivel de energía sonora de al menos un 25% respecto al nivel de fondo caracterizado por el descriptor LA90.

Los resultados indicaron que el número de eventos sonoros anómalos presentes en una ubicación está altamente correlacionado con la variabilidad de la energía sonora en la zona. Además, se evidenció que el factor cresta es un parámetro clave para estimar la magnitud del impacto acústico.

## Bases Teóricas

### Dispositivos Edge:

El incremento en la demanda de los servicios y aplicaciones en las últimas décadas del uso de la Internet,ha contribuido a un fuerte aumento de los requisitos de procesamiento y almacenamiento de datos. Son diversos,en términos de los recursos que requieren las diferentes aplicaciones y, por lo tanto, a menudo invocan soluciones a medida (Doluí y Kanti Datta, 2017).

Según Medina (2019) Edge Device, en este contexto, se refiere a elementos con capacidades limitadas que tiene su propio conjunto de recursos: CPU, memoria, almacenamiento, y red. Pueden ser Smartphone, Smartglasses, smartwatches, tablets, routers, vehículos autónomos, o cualquier dispositivo de IoT con capacidad de proceso. En este escenario, Edge Computing, representa una solución para enfrentar y aliviar la carga de procesamiento y almacenamiento en la nube, en aplicaciones y tecnologías que requieren de un ancho de banda exponencial y una baja o nula latencia.

Shi, Cao, Zhang, Li y Xu (2016) definen Edge Computing como un paradigma de computación distribuida que acerca el procesamiento y almacenamiento de datos a la fuente de generación, es decir, al "borde" de la red, en lugar de enviar toda la información a la nube centralizada. Este enfoque permite reducir la latencia, optimizar el uso del ancho de banda y mejorar la eficiencia en aplicaciones que requieren respuestas en tiempo real, como el Internet de las Cosas (IoT), la realidad aumentada, los vehículos autónomos y las ciudades inteligentes. Edge Computing se basa en dispositivos periféricos (edge devices) y nodos locales que realizan tareas de procesamiento y almacenamiento, lo que reduce la dependencia de la infraestructura centralizada de la nube.

### Raspberry pi

Según Richardson y Wallace (2016), la Raspberry Pi es un dispositivo de computación de bajo costo y alto rendimiento que ha ganado popularidad en diversos campos debido a su versatilidad y facilidad de uso. Este dispositivo, del tamaño de una tarjeta de crédito, está equipado con un procesador ARM, memoria RAM, puertos de entrada/salida y conectividad de red, lo que lo hace ideal para proyectos educativos, de automatización y desarrollo de prototipos. La Raspberry Pi es capaz de ejecutar sistemas operativos basados en Linux, lo que permite a los usuarios programar y personalizar sus aplicaciones según sus necesidades (Richardson y Wallace, 2016).

Además, Upton y Halfacree (2020) destacan que la Raspberry Pi ha evolucionado significativamente desde su lanzamiento, con modelos más potentes como la Raspberry Pi 4, que ofrece mayores capacidades de procesamiento, almacenamiento y conectividad. Este modelo incluye soporte para redes Gigabit Ethernet, puertos USB 3.0 y salidas de video de alta definición, lo que lo convierte en una herramienta poderosa para aplicaciones más exigentes, como servidores domésticos, centros multimedia y sistemas embebidos avanzados (Upton y Halfacree, 2020).

### Python

Python es un lenguaje de programación poderoso, elegante y fácil de leer, diseñado para simplificar la creación de programas mediante una sintaxis clara y estructurada. Según el documento, Python destaca por su versatilidad en aplicaciones del mundo real, su enfoque en la legibilidad del código y su capacidad para integrar paradigmas como la programación orientada a objetos y funcional. Además, es software libre con una comunidad activa y una implementación estándar consolidada (Yuill & Halpin, 2006).

### Tensorflow

Según Goldsborought (2016) TensorFlow es una biblioteca de software de deep learning de código abierto desarrollada por Google que permite definir, entrenar y desplegar modelos de machine learning mediante la representación de algoritmos como grafos computacionales.

La librería TensorFlow opera construyendo un grafo computacional en el que cada nodo representa una operación (por ejemplo, una función matemática, una transformación o una capa de una red neuronal) y cada arista transporta un tensor, es decir, un arreglo multidimensional de datos. Esta arquitectura facilita diversas optimizaciones, como la eliminación de subgrafos redundantes, y permite distribuir la ejecución de la computación a lo largo de múltiples dispositivos (CPUs, GPUs, TPUs) e incluso en entornos distribuidos. De esta manera, se optimiza tanto el uso de memoria como el rendimiento, haciendo viable el entrenamiento y despliegue de modelos complejos a gran escala. (Goldsborought, 2016).

### NestJs

Según Sabo (2020) NestJS es un framework para el desarrollo de aplicaciones del lado del servidor basado en Node.js, que se escribe en TypeScript. Proporciona una estructura modular y escalable mediante el uso de patrones modernos como la inyección de dependencias, controladores y módulos, facilitando la creación de aplicaciones backend mantenibles y robustas.

NestJS aprovecha la solidez de Node.js y Express.js, pero se diferencia al introducir un enfoque inspirado en Angular para la organización de la aplicación. Gracias a su arquitectura basada en módulos, cada parte de la aplicación se encapsula en unidades independientes que facilitan la reutilización y la escalabilidad. Además, el framework implementa un avanzado sistema de inyección de dependencias, lo que permite gestionar y suministrar las instancias de servicios de manera automática, reduciendo el acoplamiento entre componentes y promoviendo un diseño orientado a pruebas. Otro pilar fundamental de NestJS es su fuerte integración con TypeScript, lo cual aporta tipificación estática y facilita la detección temprana de errores durante el desarrollo. La utilización de decoradores en NestJS permite agregar metadatos a clases, métodos y propiedades, lo que habilita la implementación de características como interceptores, pipes y controladores para la validación y transformación de datos, así como para el manejo de rutas HTTP. Además, la herramienta Nest CLI agiliza la generación de nuevos proyectos y componentes, garantizando que se siga una estructura coherente en toda la aplicación, lo que resulta especialmente útil en proyectos complejos y colaborativos (Sabo, 2020).

### Inteligencia artificial

La Inteligencia Artificial (IA) se define como el estudio de agentes que perciben su entorno a través de sensores y actúan sobre él mediante actuadores, con el objetivo de maximizar su utilidad esperada. Según Russell y Norvig (2022), " La IA es el estudio de agentes que reciben percepciones del entorno y realizan acciones. Cada uno de estos agentes implementa una función que asigna secuencias de percepción a acciones, y cubrimos diferentes formas de representar estas funciones para lograr el mejor resultado esperado." (p. 19).

Los fundamentos de la IA se entrelazan con múltiples disciplinas. La filosofía aporta marcos éticos y lógicos, como señalan los autores: "El filósofo griego Aristóteles fue uno de los primeros en intentar codificar el "pensamiento correcto" sus silogismos proporcionaron patrones para estructuras argumentales que siempre produjeron conclusiones correctas." (p. 21). Las matemáticas y la estadística proporcionan herramientas para el razonamiento probabilístico: "La probabilidad rápidamente se convirtió en una parte invaluable de las ciencias cuantitativas, ayudando a lidiar con mediciones inciertas y teorías incompletas." (p. 26). La economía contribuye con teorías de decisión y utilidad: "La teoría de la decisión, que combina la teoría de la probabilidad con la teoría de la utilidad, proporciona un marco formal y completo para las decisiones individuales tomadas en condiciones de incertidumbre." (p. 28). La neurociencia y la psicología inspiran modelos cognitivos: "El campo interdisciplinario de la ciencia cognitiva reúne modelos informáticos de la IA y técnicas experimentales de la psicología para construir teorías precisas y comprobables de la mente humana." (p. 21). Por último, la ingeniería y la computación permiten implementar sistemas eficientes: "La historia de la IA es también la historia del diseño de arquitecturas cada vez más sofisticadas para programas de agentes." (p. 65).

La IA actual ha alcanzado hitos significativos en diversos dominios. De acuerdo a los expuesto por Russell & Norvig (2022): “Los sistemas que usan IA han alcanzado o superado el rendimiento humano en ajedrez, Go, póquer, Pac-Man, Jeopardy, detección de objetos ImageNet, reconocimiento de voz y diagnóstico de retinopatía diabética." (p. 46). En aplicaciones prácticas, destacan los vehículos autónomos: "Los vehículos de prueba de Waymo superaron la marca de 10 millones de millas recorridas en vías públicas sin sufrir accidentes graves." (p. 47), y sistemas de diagnóstico médico: "Los algoritmos de IA ahora igualan o superan a los médicos expertos en el diagnóstico de muchas afecciones, como el cáncer metastásico y las enfermedades oftálmicas." (p. 48). Además, herramientas como "Los sistemas de traducción automática ahora permiten la lectura de documentos en más de 100 idiomas, lo que genera cientos de miles de millones de palabras por día." (p. 47) evidencian su impacto global. No obstante, persisten retos éticos y técnicos, como la alineación de valores humanos y la escalabilidad en entornos complejos, que definen la frontera actual de investigación.

### Cadenas de Markov

Fundamentos teóricos Las Cadenas de Markov, nombradas en honor al matemático ruso Andrei Andreevich Markov (1856–1922), son procesos estocásticos que modelan sistemas donde el futuro depende únicamente del estado presente, sin influencia directa del pasado (Matas Soberón, s.f., p. 13). Este principio, conocido como propiedad de Markov, fue inicialmente aplicado por Markov en el análisis de secuencias de vocales y consonantes en textos literarios, como Eugene Onegin de Pushkin, sentando las bases para su uso en campos como la física, biología y ciencias sociales (Matas Soberón, s.f., p. 11). Posteriormente, estas cadenas han sido utilizadas para modelar sistemas en epidemiología, teoría de colas y dinámica de poblaciones (Bobadilla Osses, 2010, p. 5).

Una Cadena de Markov se define formalmente como un proceso estocástico {X\_n} sobre un espacio de estados S (finito o numerable), donde la probabilidad de transición al siguiente estado depende exclusivamente del estado actual. Esta propiedad se traduce en una matriz de transición P = (p\_ij), donde cada entrada p\_ij representa la probabilidad de pasar del estado i al estado j. La matriz P es estocástica, es decir, sus filas suman 1 y sus elementos son no negativos (Matas Soberón, s.f., p. 14-15). Además, en sistemas más avanzados, se pueden considerar Cadenas de Markov de tiempo continuo, las cuales presentan una relación con la teoría de semigrupos de operadores (Bobadilla Osses, 2010, p. 6).

Las cadenas de Markov pueden clasificarse en homogéneas, cuando la matriz de transición P permanece constante en el tiempo (Matas Soberón, s.f., p. 23). Un estado es recurrente si la cadena regresa a él infinitas veces, y transitorio si eventualmente lo abandona para siempre. Esto se determina mediante la representación canónica de P, que identifica clases cerradas (conjuntos de estados que no pueden abandonarse) (Matas Soberón, s.f., p. 24-25). En el contexto de sistemas biológicos, por ejemplo, una cadena de Markov puede utilizarse para modelar la propagación de una enfermedad dentro de una población, donde los estados pueden representar niveles de infección y la recurrencia indicar posibles rebrotes (Bobadilla Osses, 2010, p. 91).

Para cadenas regulares (matrices P con todas sus entradas positivas en alguna potencia), las probabilidades convergen a un estado estacionario w, único vector estocástico que satisface w = wP. Este resultado se fundamenta en el teorema de Perron-Frobenius y técnicas como la descomposición de Jordan (Matas Soberón, s.f., p. 29-30). Un caso particular de convergencia ocurre en modelos de colas, donde la distribución estacionaria describe la cantidad promedio de clientes en espera en el largo plazo (Bobadilla Osses, 2010, p. 84).

Los autores en sus respectivas publicaciones ilustran las utilidades de las Cadenas de Markov en casos reales. En fútbol, se utilizan para modelizar resultados (ganar, empatar, perder) en partidos de la liga española, calculando probabilidades de equilibrio para equipos ganadores y perdedores (Matas Soberón, s.f., p. 33-41). En póquer, se emplean para el análisis de rondas de apuestas y distribución de cartas, identificando estados recurrentes (como abandonar una partida) mediante matrices de transición (Matas Soberón, s.f., p. 42-48). En Google AdWords, se aplican para predecir el comportamiento de clics en anuncios, utilizando cadenas regulares para optimizar estrategias de marketing (Matas Soberón, s.f., p. 49-53). También se usan en el modelo de Reed-Frost en epidemiología para estudiar la propagación de enfermedades infecciosas, clasificando estados en susceptibles, infectados y recuperados (Bobadilla Osses, 2010, p. 91-107). Otro ejemplo es su aplicación en cadenas de colas, modelos de atención en sistemas de servicio como supermercados o sistemas de telecomunicaciones, donde la matriz de transición describe la dinámica de llegada y atención de clientes (Bobadilla Osses, 2010, p. 57-61).

### Agente

Según Russell y Norvig (2022), un agente es cualquier entidad que percibe su entorno a través de sensores y actúa sobre él mediante actuadores. En este sentido, un agente puede ser un ser humano, un robot o un sistema de software, siempre que cumpla con esta función de percepción y acción. Por ejemplo, un agente humano posee sensores como los ojos y oídos, y actuadores como las manos y las piernas. Un agente robótico puede incluir cámaras y sensores infrarrojos como entrada, y motores como salida. Un software, en cambio, percibe datos en forma de archivos o paquetes de red y responde modificando archivos o enviando información a otros sistemas (Russell & Norvig, 2022).

El comportamiento de un agente está determinado por su función de agente, la cual define cómo actúa en función de la secuencia de percepciones que ha experimentado. En otras palabras, un agente toma decisiones basándose en su conocimiento previo y en los datos que recibe en tiempo real. Russell y Norvig (2022) también introducen el concepto de agente racional, que es aquel que elige acciones que maximizan su desempeño según un criterio predefinido. La complejidad del entorno en el que opera el agente influye directamente en su diseño y en su capacidad de tomar decisiones óptimas (Russell & Norvig, 2022).

## Bases Legales.

El desarrollo del Sistema de Monitoreo Acústico se fundamenta en el cumplimiento de las normativas venezolanas vigentes relacionadas con la privacidad, protección de datos y seguridad informática. A continuación, se detallan los instrumentos legales aplicables:

* + - 1. Constitución de la República Bolivariana de Venezuela (1999)
* Artículo 60: Garantiza el derecho a la protección de la vida privada, intimidad, honor, propia imagen, confidencialidad y reputación. El sistema respeta este principio al no almacenar grabaciones de audio ni recopilar datos personales sensibles, limitándose al análisis acústico en tiempo real y a la generación de alertas basadas en patrones predefinidos.
* Artículo 28: Establece el derecho a la protección de datos personales. El sistema utiliza una base de datos local, sin transmisión a la nube, asegurando que la información procesada (como eventos acústicos clasificados) no permita identificar directamente a los usuarios.
  + - 1. Ley Especial contra Delitos Informáticos (2001)
* Artículo 6: Prohíbe el acceso no autorizado a sistemas informáticos. El diseño del sistema incluye medidas de autenticación y encriptación para proteger el acceso al servidor local y evitar intrusiones externas.
* Artículo 20: Sanciona la violación de la privacidad mediante la interceptación de comunicaciones. Al no grabar ni almacenar audios, el sistema evita cualquier forma de interceptación ilegítima de información personal.
  + - 1. Ley de Mensajes de Datos y Firmas Electrónicas (2001)
* Artículo 10: Establece la validez jurídica de los mensajes de datos. Aunque el sistema no genera documentos electrónicos, garantiza la integridad de los registros de eventos mediante algoritmos de hash, asegurando la autenticidad de las alertas generadas.
  + - 1. Normas COVENIN
* Norma COVENIN 2500-1: Especifica requisitos técnicos para sistemas electrónicos de seguridad. El sistema cumple con estándares de fiabilidad y funcionalidad, garantizando un monitoreo continuo sin comprometer la privacidad.
* Norma COVENIN 27001: Orienta sobre gestión de seguridad de la información. El servidor local implementa protocolos de seguridad física y lógica, como firewalls y restricciones de acceso, para proteger los datos procesados.
  + - 1. Ley Orgánica de Telecomunicaciones (2010)
* Artículo 3: Promueve el uso ético de las telecomunicaciones. Al operar en una red local sin dependencia de servicios externos, el sistema evita riesgos asociados a la transmisión de datos sensibles a través de redes públicas.

El proyecto prioriza la privacidad mediante:

* No almacenamiento de audios: Los sonidos capturados se procesan en tiempo real y se descartan inmediatamente después de su análisis.
* Base de datos local: La información técnica (como patrones de sonido y registros de alertas) se almacena en un servidor interno, sin conexión a internet, cumpliendo con el principio de confidencialidad.
* Transparencia: Los usuarios son informados sobre el funcionamiento del sistema y su finalidad, asegurando consentimiento informado conforme al artículo 60 constitucional.

# Capitulo III. Marco Metodológico

Este capítulo describe el conjunto de estrategias, técnicas y procedimientos empleados para el desarrollo del Sistema de Monitoreo Acústico para Identificar Sonidos y Generar Alertas de Emergencia. En él se detalla tanto el enfoque investigativo como las herramientas utilizadas para la recolección de datos y, fundamentalmente, la metodología de desarrollo adoptada, basada en el modelo espiral.

## Tipo de Investigación

El presente trabajo constituye una investigación de tipo proyectiva. De acuerdo con Hurtado (2000), la investigación proyectiva tiene como objetivo diseñar o crear propuestas dirigidas a resolver determinadas situaciones. Los proyectos de arquitectura e ingeniería, el diseño de maquinarias, la creación de programas de intervención social, el diseño de programas de estudio, los inventos, la elaboración de programas informáticos, entre otros, siempre que estén sustentados en un proceso de investigación, son ejemplos de investigación proyectiva. Este tipo de investigación potencia el desarrollo tecnológico (p. 325).

Basado en esta definición, el desarrollo de un sistema de monitoreo acústico para identificar sonidos y generar alertas de emergencia se adapta perfectamente a este tipo de investigación. También se conoce a las investigaciones proyectivas como proyectos factibles, entendiendo que un proyecto factible “consiste en la investigación, elaboración y desarrollo de una propuesta de un modelo operativo viable para solucionar problemas, requerimientos o necesidades…” y, además, “…el proyecto debe tener apoyo en una investigación de tipo documental, de campo o un diseño que incluya ambas modalidades” (UPEL, 2003; citado por Dubs, 2004).

El sistema propuesto en este trabajo se fundamenta en un proceso de investigación que incluye la revisión documental de conceptos como edge computing, procesamiento de señales acústicas y modelos de inteligencia artificial, así como el diseño e implementación de un prototipo funcional. Esto cumple con los requisitos de una investigación proyectiva, ya que no solo se analiza y diagnostica el problema, sino que también se desarrolla una solución tecnológica viable y aplicable en entornos reales.

## Técnicas e Instrumentos de Recolección de Datos

Según Hernández, Fernández y Baptista (2014), las técnicas de recolección de datos son procedimientos sistemáticos que permiten obtener información relevante para la investigación, mientras que los instrumentos son las herramientas específicas utilizadas para registrar dicha información.

En este estudio, se emplearon técnicas tanto documentales como experimentales. La técnica documental se utilizó para recopilar información teórica y antecedentes relacionados con el procesamiento de señales acústicas, la inteligencia artificial y los dispositivos edge computing. Para ello, se revisaron artículos científicos, libros especializados y normativas legales, utilizando instrumentos como fichas bibliográficas y matrices de análisis documental. Esta técnica permitió fundamentar teóricamente el diseño del sistema y justificar las decisiones técnicas adoptadas. Por otro lado, la técnica experimental se aplicó durante la fase de implementación y validación del sistema. Para ello, se diseñaron experimentos controlados en los que se capturaron y analizaron sonidos ambientales utilizando micrófonos y dispositivos Raspberry Pi. Los instrumentos utilizados en esta fase incluyeron software de procesamiento de señales, así como protocolos de registro de datos para documentar los resultados de las pruebas. Como señala Tamayo (2004), "la recolección de datos debe ser precisa y sistemática para garantizar que la información obtenida sea válida y confiable" (p. 145). En este sentido, se implementaron protocolos estandarizados para la captura y clasificación de sonidos, asegurando que los datos fueran consistentes y representativos de los escenarios reales en los que se espera que opere el sistema.

## Metodología de Desarrollo Utilizada

Para la realización de este trabajo se optó por el uso de la metodología Espiral como la metodología de desarrollo. Según lo propuesto por Barry Boehm en 1986, es un enfoque iterativo y flexible que combina elementos del desarrollo incremental y el prototipado, permitiendo gestionar riesgos y adaptar el sistema a medida que avanza el proyecto. Este modelo ha sido ampliamente reconocido por su capacidad para gestionar el desarrollo de sistemas complejos, ya que combina la naturaleza iterativa del prototipado con la estructura sistemática del modelo en cascada (Pressman, 2014, p. 39). En cada iteración, se pueden realizar ajustes en el plan del proyecto, permitiendo adaptar el software a las necesidades emergentes y reducir riesgos antes de que se conviertan en problemas críticos (Pressman, 2014, p. 40).

Según Boehm (1988), el Modelo Espiral se caracteriza por su estructura cíclica, en la que cada iteración incluye cuatro fases principales:

1. ***Planificación***. donde se identifican los objetivos del proyecto, los requisitos principales y las restricciones que pueden influir en su desarrollo (Boehm, 1988). Se consideran alternativas y estrategias para alcanzar estos objetivos de manera eficiente.
2. ***Análisis de riesgos***. Se identifican y evalúan los riesgos potenciales. Se definen acciones para reducir los riesgos identificados y se evalúan alternativas existentes partiendo de prototipos, simulaciones y softwares de análisis. En este ciclo, existen varios prototipos como plantillas de diseño o componentes funcionales.
3. ***Desarrollo***. se identifican los posibles factores que pueden afectar el éxito del proyecto. Se realiza una evaluación detallada de los riesgos y se proponen estrategias para su mitigación, incluyendo el uso de prototipos, simulaciones o pruebas preliminares para reducir la incertidumbre (Boehm, 1988).
4. ***Evaluación***. se revisan los productos obtenidos en la iteración actual. Esta revisión implica la participación de los interesados para verificar que el sistema cumple con los requisitos establecidos y permite realizar ajustes para la siguiente iteración. Esta fase es clave para garantizar la mejora continua del software y la adaptación a nuevas necesidades (Pressman, 2014, p. 40).

Como señala Boehm (1988), "el Modelo Espiral es particularmente adecuado para proyectos con altos niveles de incertidumbre y requisitos cambiantes, ya que permite incorporar retroalimentación continua y adaptar el diseño en función de los resultados obtenidos”. En este trabajo, esta metodología permitió gestionar eficientemente los riesgos técnicos y asegurar que el sistema cumpliera con los objetivos planteados.

# Referencias Bibliográficas

Malmberg, C., Radszuweit, D. (2021). *Real-time Audio Classification on an Edge Device - Using YAMNet and TensorFlow Lite.* Recuperado de: <https://www.diva-portal.org/smash/record.jsf?pid=diva2%3A1605037&dswid=6895>

Torija, A., Ruiz, P., Ramos-Diao, A. (2018). *Metodología para la identificación de eventos sonoros anómalos.* Recuperado de: <https://documentacion.sea-acustica.es/publicaciones/Coimbra08/id206.pdf>

Doluí, K., & Kanti Datta, S. (2017). *Comparison of Edge Computing implementations: Fog computing, cloudlet and mobile Edge Computing.* Global Internet of Things Summit (GloTS). Recuperado de: <https://doi.org/10.1109/giots.2017.8016213>

Medina, M. (2019). Edge Computing para IoT. Recuperado de: <https://openaccess.uoc.edu/bitstream/10609/91207/7/mmedinabarTFM0119memoria.pdf>

Shi, W., Cao, J., Zhang, Q., Li, Y., & Xu, L. (2016). Edge Computing: Vision and Challenges. IEEE Internet of Things Journal, 3(5), 637-646. Recuperado de: <https://doi.org/10.1109/JIOT.2016.2579198>

Richardson, M., & Wallace, S. (2016). *Getting Started with Raspberry Pi: An Introduction to the Fastest-Selling Computer in the World (4th ed.).* Maker Media, Inc. Recuperado de: <https://books.google.co.ve/books/about/Getting_Started_With_Raspberry_Pi.html?id=w4CkDAAAQBAJ&redir_esc=y>

Upton, E., & Halfacree, G. (2020). *Raspberry Pi User Guide (4th ed.)*. John Wiley & Sons. Recuperado de: <https://www.perlego.com/book/997788/raspberry-pi-user-guide-pdf>

Yuill, S., Halpin, H. (2006). *Python.* Recuperado de: <https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=1f2ee3831eebfc97bfafd514ca2abb7e2c5c86bb>

Goldsborough, P. *A Tour of TensorFlow, Proseminar Data Mining*. Technische Universität München. Recuperado de: <https://arxiv.org/pdf/1610.01178>

Sabo, M. (2020). NestJS. Završni rad. Josip Juraj Strossmayer University of Osijek, Departamento de Matemáticas. Recuperado de: <https://zir.nsk.hr/islandora/object/mathos:441>

Russell, S., & Norvig, P. (2022). *Artificial Intelligence: A Modern Approach (4th ed.)*. Pearson. Recuperado de: <http://lib.ysu.am/disciplines_bk/efdd4d1d4c2087fe1cbe03d9ced67f34.pdf>

Matas Soberón, J. J. (s.f.). *Una Introducción a las Cadenas de Markov y sus Aplicaciones*. Universitat de les Illes Balears. Recuperado de: <https://dspace.uib.es/xmlui/bitstream/handle/11201/151803/Memoria_EPSU0697.pdf?sequence=1&isAllowed=y>

Bobadilla Osses, A. (2010). *Cadenas de Markov*. Universidad de Valparaíso. Recuperado de: <https://repositoriobibliotecas.uv.cl/serveruv/api/core/bitstreams/f4406bb7-5f33-4a41-8333-7d822360a0ed/content>

Igba, J., Alemzadeh, K., Durugbo, C., & Eiriksson, E. T. (2016). Analysing RMS and peak values of vibration signals for condition monitoring of wind turbine gearboxes. Renewable Energy, 91, 90-106. <https://www.sciencedirect.com/science/article/pii/S0960148116300064?__cf_chl_tk=yBOxF8xp4UFR_pCpXdG9_RfMY7AQ31U39jXCvKI8pNs-1739718239-1.0.1.1-HGPVYiCKZYdCe_uFGG5zlj7eRTcTx3qeI6qRqZDeYKI>

Asamblea Nacional Constituyente. (1999). *Constitución de la República Bolivariana de Venezuela*. Gaceta Oficial N° 36.860. Recuperado en: <https://www.oas.org/dil/esp/constitucion_venezuela.pdf>

República Bolivariana de Venezuela. (2001). *Ley Especial contra Delitos Informáticos*. Gaceta Oficial N° 37.313. Recuperado en: <https://www.oas.org/juridico/spanish/mesicic3_ven_anexo18.pdf>

República Bolivariana de Venezuela. (2001). *Ley de Mensajes de Datos y Firmas Electrónicas*. Gaceta Oficial N° 37.348. Recuperado en: <https://www.oas.org/juridico/spanish/mesicic3_ven_anexo19.pdf>

Comisión Venezolana de Normas Industriales (COVENIN). (2005). *Norma COVENIN 2500-1: Sistemas electrónicos de seguridad – Requisitos generales*. Caracas: Fondonorma. Recuperado en: <https://www.medicinalaboraldevenezuela.com.ve/normas_covenin.html>

República Bolivariana de Venezuela. (2010). *Ley Orgánica de Telecomunicaciones*. Gaceta Oficial N° 6.013. Recuperado en: <https://www.oas.org/juridico/spanish/cyb_ven_ley_telecomunicaciones.pdf>

Hurtado, J. (2000). *Metodología de la investigación Holística*. Recuperado de: <https://ayudacontextos.wordpress.com/wp-content/uploads/2018/04/jacqueline-hurtado-de-barrera-metodologia-de-investigacion-holistica.pdf>

UPEL. (2004). *Manual de Trabajos de Grado de Especialización y Maestría y Tesis Doctorales*. Fondo Editorial de la Universidad Pedagógica Experimental Libertador

Dubs, R. (2004). *Una Estrategia Metodológica para el Proyecto Factible*. Universidad Pedagógica Experimental Libertador. Recuperado de: <https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwj-juXnzNCLAxVXxjgGHQKJAO0QFnoECBQQAQ&url=http%3A%2F%2Fhistorico.upel.edu.ve%3A81%2Frevistas%2Findex.php%2Fentretemas%2Farticle%2Fdownload%2F1032%2F363&usg=AOvVaw3YvLeNxZENzrxlilGfgPWL&opi=89978449>

Hernández, R., Fernández, C., & Baptista, P. (2014). *Metodología de la investigación.* McGraw-Hill. Recuperado de: <https://www.esup.edu.pe/wp-content/uploads/2020/12/2.%20Hernandez,%20Fernandez%20y%20Baptista-Metodología%20Investigacion%20Cientifica%206ta%20ed.pdf>

Tamayo, M. (2004). *El proceso de la investigación científica.* Limusa. Recuperado de: <https://www.gob.mx/cms/uploads/attachment/file/227860/El_proceso__de_la_investigaci_n_cient_fica_Mario_Tamayo.pdf>

Boehm, B. (1988). A spiral model of software development and enhancement. ACM SIGSOFT Software Engineering Notes, 11(4), 14-24. Recuperado en: <https://www.cse.msu.edu/~cse435/Homework/HW3/boehm.pdf>

Pressman, R. (2010). Ingeniería del software: Un enfoque práctico (7ma edicion). McGraw-Hill. Recuperado en: <https://www.academia.edu/87171392/Ingeniería_de_software_enfoque_practico_Roger_Pressman_7ma_edición?auto=download>