EDUCACIÓN
IPSUMTEC
ISSN: 2594 - 2905

SISTEMA INTELIGENTE DE MONITOREO DE VARIABLES FÍSICAS DE COLMENAS APÍCOLAS UTILIZANDO REDES NEURONALES

INTELLIGENT SYSTEM FOR MONITORING PHYSICAL VARIABLES IN BEEHIVES USING NEURAL NETWORKS

Montaño Alcalá Carlos Humberto[1], Serna Encinas María Trinidad[2*], Hernández Aguirre Fredy Alberto[3]

https://doi.org/10.61117/ipsumtec.v9i1.466

[1]Tecnológico Nacional de México/I. T. de Hermosillo, m16330894@hermosillo.tecnm.mx, https://orcid.org/0009-0006-3422-2877
[2*]Tecnológico Nacional de México/I. T. de Hermosillo, maria.sernae@hermosillo.tecnm.mx, https://orcid.org/0000-0002-2020-791X
[3]Tecnológico Nacional de México/I. T. de Hermosillo, fredy.hernandeza@hermosillo.tecnm.mx, https://ocid.org/0000-0001-9208-5299

Resumen - La apicultura es una actividad esencial tanto para la producción de miel como para la polinización de plantas. En México, la apicultura juega un papel significativo, siendo el noveno productor de miel a nivel mundial. Para lograr mantener las colmenas en un buen estado de salud y con un alto nivel de producción, es necesario mantener un monitoreo constante, tradicionalmente este proceso se realiza mediante revisiones manuales de la colmena en periodos de 8 a 15 días con el objetivo de detectar problemas o enfermedades. Conforme aumentan el número de colmenas, el monitoreo manual se vuelve ineficiente, tardado y logísticamente complicado, con la posibilidad de provocar que algunas colmenas no sean revisadas en el periodo correcto. Este artículo presenta un sistema que monitorea las colmenas utilizando tecnologías de internet de las cosas, integrando sensores que monitorean variables como temperatura, humedad peso y señales de audio de la colmena con un algoritmo de reconocimiento de patrones, utilizado para detectar eventos de interés para el apicultor. El sistema también incorpora una aplicación móvil con la capacidad de visualizar las alertas y las mediciones de los sensores. El propósito de este proyecto es proporcionar a los apicultores con datos en tiempo real sobre el estado de la colmena, así como de apoyar a la logística y a la toma de decisiones.

***Palabras Clave*:** Colmenas apícolas, IoT, Redes neuronales, Sistema de monitoreo, Variables físicas.

Abstract: Beekeeping activities are important not just for honey but also for pollinating plants. In Mexico, it is a relevant economic activity, and the country is placed as the ninth biggest honey producer on the world. In order to keep the hives in good health and for effective output there is a need for constant monitoring. Traditionally, this process is made by checking hives manually every 8 to 15 days to see any problems or sickness. As the number of hives rises, this kind of monitoring becomes inefficient, time consuming and more difficult in planning and can even mean some hives are not checked at the right time. This article presents a system that checks hives using Internet of Things technology for monitoring. It integrates sensors that watch over variables such as the hive temperature, wetness, weight and sound; and a pattern recognition algorithm to detect important events needing attention from the beekeeper. The system includes a mobile app which can display the alerts and the sensors output. The purpose of this project is to help beekeepers get instant data about their hive status, aiding their choices and management.

Keywords: Beehives, IoT, Neural networks, Monitoring system, Physical variables.

INTRODUCCIÓN

La apicultura es una actividad económica orientada a la explotación de colmenas apícolas y al mantenimiento de su estado sanitario, para lo cual es necesario incorporar a la apicultura una metodología técnica, que involucra el manejo correcto de reinas, el control de enfermedades, la eliminación de parásitos y la optimización de la producción [1]. Asegurar la salud y producción eficiente de las colmenas requiere contar con un monitoreo constante, mediante inspecciones realizadas por el apicultor en intervalos de 8 a 15 días, con el objetivo de buscar anomalías y enfermedades [2].

Además de su función productiva, las colmenas contribuyen a la polinización de la vegetación circundante; esto tiene relevancia como actividad económica, ya que cerca del 75% de los cultivos alimentarios dependen de la acción polinizadora de las abejas [3].

Actualmente, se cuenta con soluciones tecnológicas enfocadas al monitoreo de colmenas mediante el uso de sistemas de internet de las cosas (IoT). El enfoque principal de estos sistemas es el monitoreo de parámetros físicos como la temperatura, la humedad, el peso y la recopilación de señales acústicas, con el objetivo de utilizar estos datos en flujos de análisis, complementados con algoritmos de inteligencia artificial (IA) [4],[5], [6], [7]. La solución propuesta en este articulo consiste en el desarrollo de un sistema inteligente basado en un esquema IoT que

,Recepción: 16/03/2026 | Aceptación: 18/05/2026 | Publicación: 22/06/2026

recolecta datos a través de nodos de sensores, que después son transmitidos a un servidor para su procesamiento, almacenamiento y visualización. Adicionalmente, se incluye en el sistema un módulo de alerta basado en redes neuronales, que tiene como objetivo la detección temprana de eventos relevantes para el manejo de la colmena.

DESARROLLO

Metodología

El desarrollo del sistema consiste en cinco fases. La primera consistió en la revisión del estado del arte en apicultura, desarrollo móvil, backend, IoT y reconocimiento de patrones; a través de artículos de congreso, revistas, tesis y estudios previos relevantes.

La segunda fase consistió en identificar los componentes y las características requeridas para el diseño y modelado del sistema de monitoreo, incluyendo la selección de sensores, fuentes de energía y el controlador.

La tercera fase consistió en el desarrollo del sistema de procesamiento, incluyendo el procesamiento de datos provenientes del sistema de monitoreo, el reconocimiento de patrones y la persistencia de datos.

La cuarta fase consistió en implementar la aplicación móvil, que tiene como objetivo procesar y visualizar la información proveniente del servicio, así como de mostrar las alertas detectadas.

Finalmente, la quinta fase consistió de la realización de pruebas integrales del sistema, así como del análisis de los resultados y el ajuste del sistema según sea requerido.

Algoritmo del microcontrolador del sistema

El algoritmo del microcontrolador mostrado en la figura 1 representa el ciclo del nodo de monitoreo. El ciclo operativo se ejecuta en un microcontrolador ESP32.

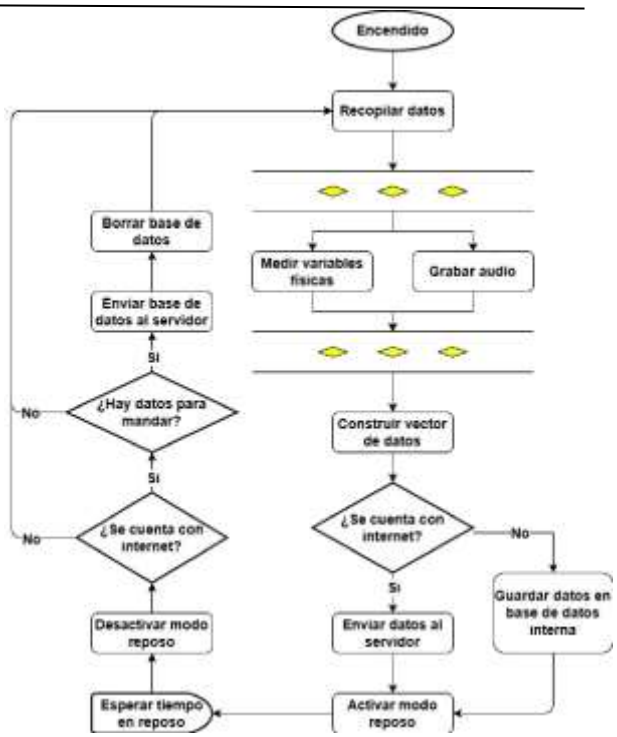

Figura 1. *Algoritmo del microcontrolador.*
Fuente. *Elaboración propia (2026).*

El flujo está estructurado como un ciclo que coordina la adquisición de datos y la sincronización con el servidor, sujeto a la disponibilidad de la conexión. El diseño incluye almacenamiento local para tolerar fallas de red, así como mecanismos de ahorro de energía que consisten en alternar entre estados de trabajo y de baja energía. Los elementos gráficos usados y su funcionamiento dentro del proceso se describen a continuación:

- Inicio (óvalo). Representa el punto de inicio del proceso.
- Decisión (rombo). Nodo en el cual el flujo se divide basado en una condición, solamente se selecciona una salida.
- Espera (rectángulo con un lado redondeado). Indica una pausa en el proceso.
- Ejecución paralela (nodos entre dos rectángulos con rombos amarillos). Sección donde múltiples nodos son ejecutados simultáneamente.

Modelos de Datos

La capa de persistencia está estructurada en 3 bases de datos funcionales, *guard-database*, *nest-database*, y *mind-database*; diseñadas bajo el principio de separación de responsabilidades.

La base de datos *nest-database* guarda información relacionada a la estructura del sistema y mediciones de los sensores, permitiendo el monitoreo del estado de cada colmena.

,Recepción: 16/03/2026 | Aceptación: 18/05/2026 | Publicación: 22/06/2026

La base de datos *guard-database* administra las identidades digitales y los tokens de acceso.

La base de datos *mind-database* agrupa datos derivados del procesamiento de señales acústicas, permitiendo su uso con modelos de aprendizaje máquina.

La figura 2 presenta el modelo de datos completo del sistema propuesto.

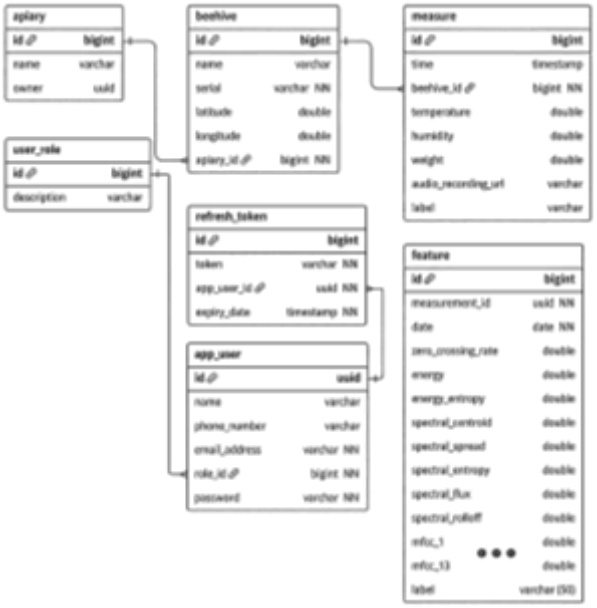

Figura 2. *Modelo de datos completo del sistema.*
Fuente. *Elaboración propia (2026).*

Base de datos de seguridad: *guard-database*

- *user_role*, define los roles disponibles en el sistema.
- *app_user*, representa los usuarios registrados y su asociación con un rol especifico.
- *refresh_token*, administra las sesiones activas mediante tokens asociados a cada usuario, permitiendo la renovación o revocación de sesiones.

Base de datos de colmenares: *nest-database*

- *apiary*, modela las agrupaciones de colmenas físicas, permitiendo su organización por ubicación o dueño.
- *Beehive*, representa cada colmena dentro de un apiario, es la unidad a monitorear.
- *Measure*, guarda las variables físicas capturadas de cada colmena.

Base de datos de características: *mind-database*

- *feature*, guarda las características extraídas de las señales acústicas.

Arquitectura del software

El sistema está estructurado en 3 componentes principales: el nodo de sensores, los servicios backend operacionales, el servicio de procesamiento de audio y la aplicación móvil. En la figura 3 se muestra la arquitectura completa del sistema.

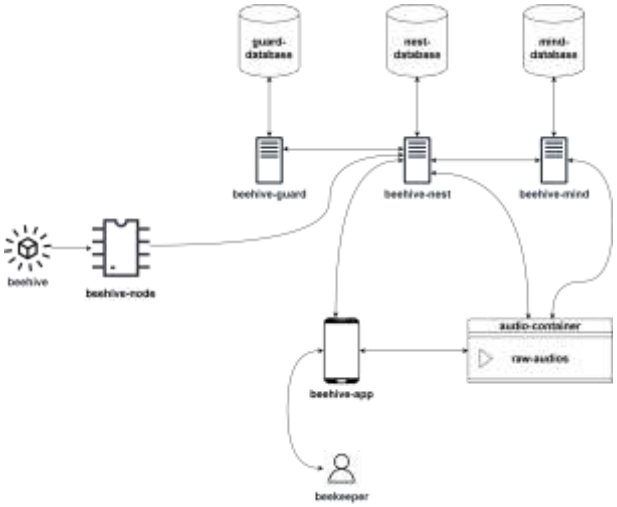

Figura 3. *Arquitectura lógica del sistema propuesto.*
Fuente. *Elaboración propia (2026).*

Nodo de sensores (beehive-node). Componente basado en microcontroladores y responsable de la adquisición de datos de la colmena; integrando sensores para medir temperatura, humedad, peso y señales acústicas.

Servicios backend:

- *beehive-guard*: responsable de administrar la seguridad del sistema, incluyendo autenticación y control de acceso.
- *beehive-nest*: responsable de administrar la información operacional del sistema, incluyendo colmenas, mediciones y apiarios.
- *beehive-mind*: servidor dedicado a extraer las características del audio incluido con cada medición, así como de ejecutar la inferencia por medio del modelo implementado.

Contenedor de audio (audio-container): La persistencia de las grabaciones de audio, se implementa mediante un servicio de almacenamiento de objetos compatible con la API S3. Bajo este enfoque, los archivos se organizan en contenedores lógicos denominados buckets. Para el entorno de pruebas se empleó MinIO como una implementación local del estándar S3 [8].

Aplicación móvil (beehive-app): Es la interfaz a la que tiene acceso el apicultor, para monitorear el estado de las colmenas en tiempo real; así como recibir notificaciones. La capacidad de visualizar en tiempo real del estado de las colmenas se logra mediante solicitudes periódicas al servidor.

A continuación, se describen los flujos principales del sistema:

1. El nodo de sensores adquiere data de la colmena y la transmite al servicio beehive-nest.
2. La aplicación móvil realiza solicitudes periódicas a beehive-nest para obtener los datos más recientes.
3. El servicio beehive-nest manda la grabación de audio al contenedor de audio.
4. El servicio beehive-mind procesa las señales acústicas, extrae las características y clasifica cada grabación como anómala o normal. El resultado se guarda en la base de datos.
5. El servicio beehive-guard administra los mecanismos de autorización aplicados a todas las interacciones entre la aplicación móvil y el sistema.

La tabla 1 presenta los componentes principales de la arquitectura del sistema propuesto.

Tabla 1. *Componentes principales de la arquitectura del sistema.*

Componente	Función	Persistencia
beehive-node	Captura de datos de sensores	Micro SD
beehive-guard	Autenticación y seguridad	PostgreSQL (guard-database)
beehive-nest	Gestión de metadatos y registros	PostgreSQL (nest-database)
beehive-mind	Análisis de audio con IA	PostgreSQL (mind-database)
audio-container	Almacenamiento de audios crudos	S3/MinIO (raw-audios)
beehive-app	Interfaz para el apicultor	Persistencia local en dispositivo móvil

Fuente. *Elaboración propia (2026).*

Flujo de la Aplicación: La aplicación móvil actúa como punto de entrada al sistema, permitiendo a los usuarios registrarse, autenticarse, acceder a información y mandar solicitudes operacionales a los servicios. La figura 4 muestra la pantalla principal de la aplicación.

Figura 4. *Pantalla principal.*
Fuente. *Elaboración propia (2026).*

Como ya se mencionó, la figura 5 muestra la pantalla principal, ésta es accesible inmediatamente después de iniciar sesión y, mediante ésta, es posible visualizar la información de usuario, administrar colmenas asociadas y cerrar sesión.

El proceso de registro requiere que el usuario ingrese una contraseña, un correo electrónico, un número de teléfono y un nombre de usuario. Una vez hecho esto, es posible iniciar sesión con estos datos.

Para añadir un apiario nuevo, el usuario debe ir a la sección de *Beehives* desde la pantalla principal, usar el botón de agregar "+" y seleccionar la opción de Apiario. Mientras que, para registrar una nueva colmena, el usuario debe de estar dentro de un apiario disponible y seleccionar la opción de *Add a new beehive* o utilizar el botón de agregar "+" y seleccionar la opción *Beehive.*

Tras la validación del dispositivo por parte del sistema, el estado de la colmena puede consultarse directamente desde la interfaz donde se listan los colmenares.

En la sección de monitoreo de la colmena se presentan los parámetros principales mediante indicadores visuales, cuyo color varía dinámicamente en función de sus valores.

- **Humedad.** Representada por un ícono de gota; el indicador adopta color amarillo cuando el valor se encuentra por debajo del rango recomendado y azul cuando lo supera.
- **Temperatura.** Mostrada mediante un indicador que cambia a azul si el valor está por debajo del rango establecido y a rojo cuando lo excede.
- **Peso.** Desplegado como un valor numérico constante, sin variación visual asociada a su magnitud.

- **Audio.** Representado por un ícono triangular, que permite reproducir la grabación más reciente vinculada a la colmena.
- **Detalles.** Identificado con un ícono circular con una "I", el cual dirige al usuario a la vista detallada de la colmena al ser seleccionado.

Las figuras 5 y 6 muestran una vista de detalles de la colmena.

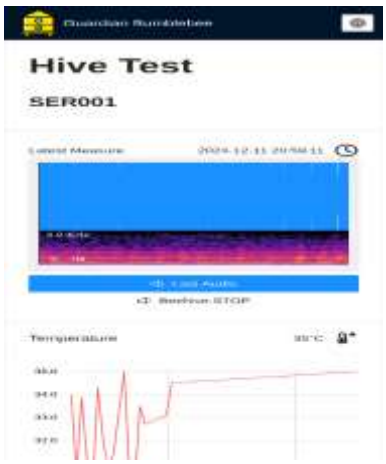

Figura 5. *Espectrograma del audio recopilado.*
Fuente. *Elaboración propia (2026).*

Figura 6. *Últimas mediciones recopiladas por el nodo de sensores.*
Fuente. *Elaboración propia (2026).*

Las figuras anteriores muestran la vista detallada de la colmena, donde se presenta el espectrograma correspondiente a la señal de audio capturada, junto con la opción de reproducirla. Asimismo, se despliegan las mediciones más recientes, obtenidas por el nodo de sensores y su comportamiento temporal, a través de una representación gráfica.

Características y clasificación del audio

Características de audio: Se describen los cálculos y el procedimiento algorítmico a implementar en Python. Se considera una señal de audio monofónica con frecuencia de muestreo **sr**; en el caso de audio estéreo, el procesamiento se realiza de forma independiente por canal. Los parámetros por defecto son n_fft = 2048, hop_length = 512 y una ventana de tipo Hann [9].

Zero Crossing Rate (ZCR)
Algoritmo [10], [11]:

1. Segmentar la señal continua en ventanas de longitud fija, por ejemplo, de 2048 muestras.
2. En cada ventana, evaluar el cambio de signo entre muestras consecutivas; cada transición de positivo a negativo, o de negativo a positivo, se cuenta como un cruce por cero.
3. Acumular el total de cruces dentro de la ventana y normalizarlo con respecto a su longitud, para obtener una métrica comparable entre señales distintas.
4. Analizar la secuencia temporal de estas tasas para detectar regiones de alta actividad, por lo general asociadas con ruido o con componentes no sonoros del habla.

Energía
Algoritmo [10], [12], [13]:

1. Dividir la señal en ventanas de duración definida.
2. En cada ventana, calcular el cuadrado de cada muestra para cuantificar la magnitud de la señal.
3. Sumar los valores obtenidos dentro de la ventana; un valor mayor indica una mayor intensidad acústica en ese intervalo.

Entropía de energía
Algoritmo [12], [14], [15]:

1. Para cada ventana de la señal, subdividirla en Q segmentos consecutivos de menor tamaño.
2. Calcular la energía de cada segmento y normalizar esos valores, para obtener proporciones cuya suma sea igual a 1.
3. Aplicar la entropía de Shannon sobre dichas proporciones; valores altos indican una distribución uniforme de la energía, mientras que valores bajos reflejan su concentración en pocos segmentos.

Centroide espectral
Algoritmo [10], [11], [15]:

1. Obtener el espectro de magnitud de cada ventana de la señal.
2. Ponderar cada componente espectral por su frecuencia correspondiente y calcular el promedio resultante, lo que permite estimar el centro de masa del espectro.

Dispersión espectral
Algoritmo [11], [15]:

1. Determinar el centroide espectral.
2. Calcular la desviación estándar de las frecuencias, ponderada por sus magnitudes espectrales, para cuantificar la dispersión o ancho del espectro.

Entropía espectral
Algoritmo [10], [15], [16]:

1. Obtener el espectro de magnitud y normalizar sus componentes para construir una distribución.
2. Calcular la entropía de Shannon sobre dicha distribución; valores altos indican una distribución uniforme de la energía en el espectro, mientras que valores bajos reflejan su concentración en un número reducido de bandas de frecuencia.

Flujo espectral
Algoritmo [10], [17]:

1. Calcular los espectros de magnitud de dos ventanas consecutivas y normalizarlos.
2. Obtener la diferencia entre el espectro actual y el anterior, conservando solo los incrementos positivos.
3. Elevar al cuadrado estas diferencias y acumular el resultado; valores altos indican transiciones abruptas en el contenido espectral, asociadas con cambios de timbre o ataques sonoros.

Rolloff espectral
Algoritmo [11], [15]:

1. Ordenar las magnitudes en función de la frecuencia ascendente.
2. Encontrar la frecuencia donde la suma acumulada alcanza una fracción fija, por ejemplo, 85%, de la energía total.

MFCC (Mel Frequency Cepstral Coefficients)
Algoritmo [10], [18], [19], [20]:

1. Segmentación y ventana: dividir la señal en tramas de tamaño fijo y aplicar una ventana para reducir discontinuidades en el análisis espectral.
2. Transformada y potencia: obtener el espectro de cada trama y calcular su energía para representar cómo se distribuye la señal en frecuencia.
3. Banco de filtros Mel: proyectar el espectro sobre un conjunto de filtros en escala Mel, con el fin de aproximar la forma en que el oído humano percibe el sonido.
4. Compresión logarítmica: aplicar una transformación logarítmica, para reducir el rango dinámico de la señal.
5. DCT: transformar los valores obtenidos para generar un conjunto pequeño de coeficientes, que describe la forma general del espectro.

Clasificación de audio

Las figuras siguientes presentan una comparación entre dos espectrogramas obtenidos de colmenas diferentes. En la figura 7 se muestra una colmena en estado normal, mientras que en la figura 8 se observa una colmena sin abeja reina. Las variaciones en la distribución de frecuencias y en la configuración del espectro indican la presencia de patrones acústicos diferenciables, lo que permite abordar el problema como una tarea de clasificación [21].

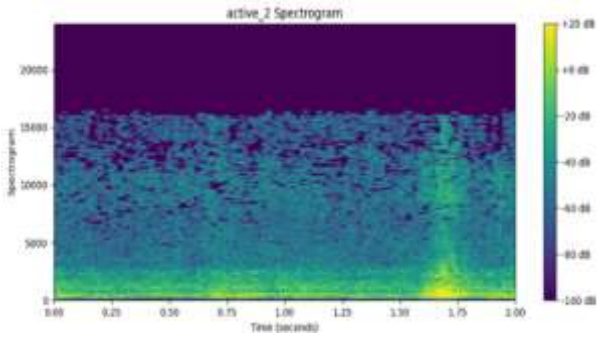

Figura 7. *Espectrograma de audio de una colmena en condición normal.*
Fuente. *Elaboración propia (2026).*

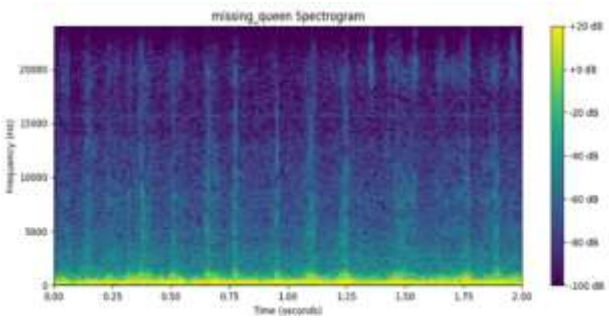

Figura 8. *Espectrograma de audio de una colmena sin reina.*
Fuente. *Elaboración propia (2026).*

A partir de las diferencias identificadas en las señales acústicas, se establece la viabilidad de su discriminación. Con base en este supuesto, se diseña e implementa una red neuronal que recibe como entrada un vector de características previamente definidas y realiza la clasificación del audio en categorías normal o anómalo, considerando condiciones asociadas al estrés de la colonia.

Conjunto de datos inicial

El modelado de audio se basa en la integración de tres conjuntos de datos públicos, que incluyen grabaciones de colmenas bajo distintas condiciones, ubicaciones y dispositivos de captura, con el fin de construir un conjunto representativo.

- **Open Source Beehives Project–Audio Database as of 2-21-17.** Contiene grabaciones organizadas por estado de la colmena y ubicación; los audios Active se consideran normales y el resto (Missing Queen, Pre-Swarm, Queen Hatching, Sick-Varroa, Swarm) como anómalos [22].
- **Queen Loss Event in Africanized Honeybee Colony**. Incluye vectores de características asociados a eventos de pérdida de reina; QR se etiqueta como normal y QL como anómalo [23].
- **Beehive buzz anomalies**. Contiene segmentos clasificados en varias categorías; se descartan

,Recepción: 16/03/2026 | Aceptación: 18/05/2026 | Publicación: 22/06/2026

los No es abeja, mientras que Abeja y Reina desaparecida se asignan como normal y anómalo, respectivamente [24].

Una vez integrados y etiquetados los datos de audio y los vectores existentes, se realizó la extracción de características a partir de segmentos de 1 segundo por grabación. Las características obtenidas, se combinaron con los vectores previamente disponibles para conformar un único conjunto de datos, en el cual cada instancia está representada por un vector de características y una etiqueta, que indica si corresponde a una condición normal o anómala. La figura 9 muestra la distribución entre ambas clases.

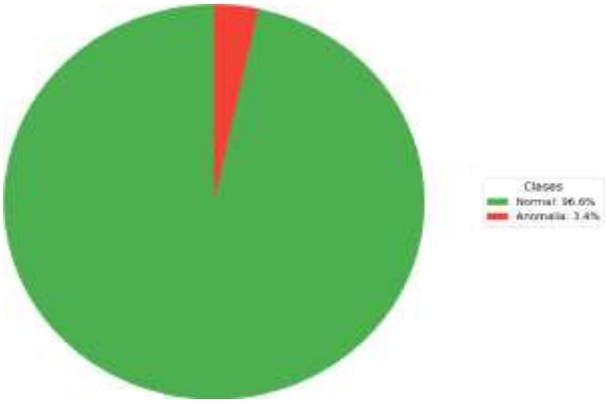

Figura 9. *Proporción anómalo-normal del conjunto de datos inicial.*
Fuente. *Elaboración propia (2026).*

Como se puede observar en la figura 10, la proporción entre segmentos anómalos y normales presenta un desbalance significativo. En este contexto, un modelo que clasifique todas las instancias como *normales* obtendría una precisión del 96%; lo que indica que esta métrica no es válida para la evaluación de desempeño del modelo, debido a que no representa su capacidad de detectar anomalías. Tomando eso en cuenta, y dado que la detección de anomalías es el objetivo principal, el proceso de entrenamiento se enfocó en optimizar la métrica F1 exclusivamente para la clase anómala. Para este propósito, se calcula la métrica F1 sobre la clase anómala, una vez terminada cada uno de los experimentos, permitiendo almacenar la métrica como resultado de éste.

Diseño de la red neuronal

El modelo fue entrenado utilizando el 80% del set de datos, mientras que el 20% restante se utilizó para validación. La selección de hiperparámetros fue realizada utilizando búsqueda aleatoria, evaluando múltiples configuraciones generadas dentro de rangos predefinidos. En cada iteración, el modelo es entrenado con el set de entrenamiento y evaluado con el set de validación, este proceso es repetido iterativamente, ajustando los rangos de búsqueda basándose en los resultados de experimentos previos y priorizando configuraciones con los mayores valores de F1 sobre la clase anómala.

La tabla 2 muestra los resultados correspondientes a los seis experimentos.

Tabla 2. *Resumen de los subconjuntos de mejores pruebas utilizados en cada experimento.*

Número de Experimento	Mejores Iteraciones	Iteraciones totales	Porcentaje (%)
1	20	200	10.00
2	20	200	10.00
3	60	600	10.00
4	17	1743	0.98
5	15	1557	0.96
6	15	1500	1.00
Total	**147**	**5800**	**2.53**

Fuente. *Elaboración propia (2026).*

La implementación completa del sistema se encuentra disponible en un repositorio público [25].

DISCUSIÓN Y ANÁLISIS DE RESULTADOS

La tabla 3 presenta los rangos máximos y mínimos de hiperparámetros obtenidos en los experimentos 1 a 3, considerando un total de 15 posibles hiperparámetros en la arquitectura de la red neuronal.

Tabla 3. *Rangos de hiperparámetros de los experimentos 1 a 3.*

Hiperparámetro	E1 min	E1 max	E2 min	E2 max	E3 min	E3 max
layers	2	2	4	4	2	4
units1	32	128	32	128	32	128
dropout1	0.2	0.5	0.2	0.4	0.2	0.4
units2	16	64	16	64	16	56
dropout2	0.2	0.5	0.2	0.4	0.2	0.4
units3	-	-	8	32	8	28
dropout3	-	-	0.2	0.5	0.2	0.4
units4	-	-	4	16	4	16
dropout4	-	-	0.2	0.6	0.2	0.5
loss_fn	focal	tversky	focal	tversky	focal	focal
focal_gamma	1.0	5.0	1.0	5.0	3.0	5.0
focal_alpha	0.1	0.9	0.1	0.9	0.2	0.5
tversky_alpha	0.1	0.8	0.1	0.08	0.15	0.55
tversky_beta	0.1	0.9	0.15	0.08	0.15	0.65

,Recepción: 16/03/2026 | Aceptación: 18/05/2026 | Publicación: 22/06/2026

lr	0.00020 8	0.008327	0.00 0258	0.00 3646	0.00 0312	0.00 3495

Fuente. *Elaboración propia (2026).*

Para comenzar, en el experimento 1 se iteraron arquitecturas limitadas a 2 capas, mientras que en el experimento 2 solamente se evaluaron arquitecturas de 4 capas, debido a los resultados variados, se añadió el número de capas como un hiperparámetro dinámico para los experimentos posteriores. Adicionalmente, entre los mejores resultados del experimento 3, solamente se observó la función de pérdida focal, por lo que fue la única que se consideró para los experimentos posteriores.

La tabla 4 presenta los rangos máximos y mínimos de hiperparámetros obtenidos en los experimentos 4 a 6, considerando un total de 12 posibles hiperparámetros en la arquitectura de la red neuronal.

Tabla 4. *Rangos de hiperparámetros de los experimentos 4 a 6.*

Hiper-pará-metro	E4 min	E4 max	E5 min	E5 max	E6 min	E6 max
layers	2	4	4	4	4	4
units1	32	128	32	128	36	124
dro-pout1	0.2	0.4	0.2	0.4	0.21	0.4
units2	16	56	16	56	30	54
dro-pout2	0.2	0.4	0.2	0.4	0.24	0.4
units3	8	28	8	26	8	25
dro-pout3	0.2	0.4	0.2	0.4	0.22	0.4
units4	4	16	5	15	5	15
dro-pout4	0.2	0.4	0.2	0.4	0.2	0.37
fo-cal_gam ma	4	5.0	4.0	5.0	4.1	4.9
fo-cal_alph a	0.2	0.4	0.2	0.4	0.21	0.39
lr	0.00040 7	0.003260	0.00 0497	0.00 2795	0.00 0549	0.00 2544

Fuente. *Elaboración propia (2026).*

A partir de los resultados obtenidos, se puede observar que los rangos óptimos, derivados del proceso iterativo de los 6 experimentos ejecutados, terminan por converger en un set de hiperparámetros estable. En la tabla 5 se pueden observar los resultados de las métricas de los 6 experimentos.

Tabla 5. Rangos de métricas obtenidas en las pruebas 1 a 6.

Experimento	Precisión (min)	Precisión (max)	f1 (min)	f1 (max)
1	0.9091	0.9963	0.4271	0.8445
2	0.9389	0.9964	0.4731	0.9470
3	0.9303	0.9964	0.4073	0.9467
4	0.9887	0.9967	0.8565	0.9507
5	0.9910	0.9967	0.8777	0.9518
6	0.9933	0.9969	0.9024	0.9546

Fuente. *Elaboración propia (2026).*

La tabla 5 presenta los intervalos máximos y mínimos, correspondientes a las métricas de precisión y F1, obtenidos entre el total de modelos iterados en cada experimento, destacando que el experimento 6 alcanza los valores más altos en ambas métricas, demostrando el avance logrado en cada experimento.

En la tabla 6 se muestran los hiperparámetros encontrados dentro del experimento 6.

Tabla 6. *Mejor configuración de hiperparámetros obtenida en la búsqueda.*

Hiperparámetro	Valor
units1	88
dropout1	0.22
units2	48
dropout2	0.24
units3	13
dropout3	0.22
units4	8
dropout4	0.35
focal_gamma	4.65
focal_alpha	0.24
lr	0.001043400598498229

Fuente. *Elaboración propia (2026).*

En las Tablas 3, 4 y 6, se listan las variables empleadas en cada sesión de experimentación. Éstas corresponden directamente a atributos definidos en el código utilizado para ejecutar las iteraciones. A continuación, se describe su significado.

- **layers**: número de capas ocultas en la red.
- **units[n]:** número de neuronas en la capa oculta n (1 a 4).
- **dropout[n]**: proporción de neuronas desactivadas en la capa n (1 a 4).
- **loss_fn**: función de pérdida empleada durante el entrenamiento.
- **focal_gamma**: parámetro gamma de la función de pérdida focal.
- **focal_alpha**: parámetro alpha de la función de pérdida focal.
- **tversky_alpha**: parámetro alpha del índice de Tversky.

,Recepción: 16/03/2026 | Aceptación: 18/05/2026 | Publicación: 22/06/2026

- **tversky_beta**: parámetro beta del índice de Tversky.
- **lr**: tasa de aprendizaje del optimizador.

La figura 10 presenta la matriz de confusión del modelo seleccionado en el experimento 6. En el conjunto de validación, dicho modelo obtiene una precisión de 99.6921% y un valor máximo de F1 de 0.954607 para la clase anómala.

Figura 10. *Matriz de confusión de la red neuronal seleccionada sobre los datos de validación.*
Fuente. *Elaboración propia (2026).*

En el conjunto de validación, el modelo alcanza una precisión de 99.6921% y un valor máximo de F1 de 0.954607 en la clase anómala.

Al comparar los resultados obtenidos por el sistema propuesto con investigaciones previas en el área del monitoreo de colmenas, se identifican diferencias notables en el método y alcance de los sistemas propuestos. El sistema embebido propuesto en [5], basado en la utilización de un microcontrolador ESP8266 y un sensor SHT21, se enfoca en el monitoreo eficiente de la temperatura y la humedad, priorizando principalmente la optimización del consumo de energía. Sin embargo, su alcance no contempla ningún tipo de análisis a los datos obtenidos, dejando esto como una posible línea de investigación a futuro.

De la misma manera, la propuesta vista en [7] se enfoca en el diseño de un sistema capaz de soportar condiciones extremas de temperatura (-20 a 60 °C), así como de mantener una alta autonomía energética, mediante el uso de supercapacitores y paneles solares. Similar a la propuesta anterior, el enfoque se basa en la adquisición y transmisión de datos, sin contemplar su posterior análisis.

Finalmente, el sistema propuesto en [26] incorpora un componente de análisis automatizado, aplicando técnicas de extracción de características mediante MFCC y LPC, junto con clasificación basada en modelos de aprendizaje maquina HMM/GMM para la detección de enjambres a partir de datos acústicos. Este enfoque demuestra la viabilidad del procesamiento inteligente en el contexto apícola. El presente trabajo se ubica en una línea similar en cuanto a la integración de análisis automatizado, explorando el uso de redes neuronales para la detección de eventos anómalos en la colmena.

CONCLUSIONES

Se desarrolló una arquitectura IoT basada en microcontrolador, para la adquisición de mediciones y su almacenamiento en una infraestructura en la nube. Como parte del sistema, se diseñó un circuito electrónico que integra sensores de humedad, peso y temperatura, así como un módulo de captura de audio y mecanismos de persistencia de datos.

En el plano de software, se implementó un servicio en la nube orientado a la gestión de colmenares, colmenas y mediciones, estructurado mediante entidades y relaciones operativas. Adicionalmente, se desarrolló un servicio de manejo y autenticación de usuarios, así como herramientas para explorar y evaluar arquitecturas con enfoque en ajuste de hiperparámetros y optimización del desempeño del modelo.

Se implementó un servicio en la nube para la extracción de vectores de características de señales de audio y su clasificación utilizando una red neuronal. También se desarrolló una aplicación móvil como interfaz del sistema, permitiendo la visualización de los datos y la interacción con el resto de los módulos.

Se implementaron mecanismos de alerta basados en límites de los parámetros monitoreados, proveyendo a los usuarios de notificaciones directas.

Por último, mediante la implementación del modelo correspondiente al experimento 6, se obtuvo una precisión máxima de 99.6921 % en la clasificación de eventos anómalos. Sin embargo, debido al desbalance del conjunto de datos, la métrica más relevante en el conjunto de validación es el F1 de la clase anómala, con un valor de 0.954607. Este resultado adquiere relevancia considerando la variabilidad inherente de las señales acústicas analizadas.

Como líneas de investigación futuras, se plantean dos direcciones principales. La primera consiste en la ampliación del conjunto de datos mediante campañas de muestreo controlado en colmenas de producción, correlacionando las señales acústicas con observaciones directas del estado de la colmena, así como la incorporación de sensores adicionales como vibración, CO2 y conteo de abejas. Esto permitiría capturar la variabilidad asociada al clima, a la genética de las abejas y a la dinámica de col-

mena; además de habilitar la distinción entre distintos tipos de eventos anómalos como pérdida de reina, el estrés térmico, los depredadores y el enjambrado.

La segunda línea contempla la optimización del modelo de clasificación, mediante la implementación de aprendizaje continuo, que permita adaptar el modelo a nuevas condiciones de campo sin re-entrenarlo desde cero, así como la ejecución de múltiples modelos en paralelo bajo un esquema de votación, para incrementar la robustez en la detección de eventos anómalos.

BIBLIOGRAFÍA

[1] J. M. Flores and J. A. Ruiz, "Principios básicos de la apicultura moderna," *Revista de Ciencias Agropecuarias*, vol. 38, no. 2, 2021, doi: 10.22201/fesz.23958723e.2021.2.82.

[2] A. Correa Ben\'{\i}tez, I. Vásquez Valencia, N. Peña Haaz, L. León Luna, and R. Anguiano Báez, "Buenas prácticas pecuarias en la producción primaria de miel," 2018. [Online]. Available: https://atlas-abejas.agricultura.gob.mx/pdfs/Manual_BPP_en_la_Producci_n_primaria_de_Miel_octubre_2018.pdf

[3] A.-M. Klein *et al.*, "Importance of pollinators in changing landscapes for world crops," *Proceedings of the Royal Society B: Biological Sciences*, vol. 274, no. 1608, 2007, doi: 10.1098/rspb.2006.3721.

[4] S. Cecchi, S. Spinsante, A. Terenzi, and S. Orcioni, "A Smart Sensor-Based Measurement System for Advanced Bee Hive Monitoring," *Sensors*, vol. 20, no. 9, p. 2726, May 2020, doi: 10.3390/s20092726.

[5] M. Vidrascu and D. Svasta, "Embedded Software for IoT Bee Hive Monitoring Node," 2017.

[6] M. Vidrascu and D. Svasta, "Maintenance-free IoT Gateway Design for Bee Hive Monitoring," 2017.

[7] M. Vidrascu, P. Svasta, and M. Vladescu, "High reliability wireless sensor node for bee hive monitoring," *Sensors and Actuators*, pp. 134–138, Mar. 2016, doi: 10.1109/SIITME.2016.7777262.

[8] "MinIO High Performance Object Storage — MinIO Object Storage (AGPLv3)." Accessed: Mar. 19, 2026. [Online]. Available: https://miniodocs.tf.fo/

[9] B. McFee *et al.*, "librosa: Audio and Music Signal Analysis in Python," *J. Open Source Softw.*, 2015, doi: 10.21105/joss.00022.

[10] M. Müller, *Fundamentals of Music Processing*. Springer, 2015. Doi: 10.1007/978-3-319-21945-5.

[11] G. Tzanetakis and P. Cook, "Musical Genre Classification of Audio Signals," *IEEE Transactions on Speech and Audio Processing*, vol. 10, no. 5, 2002, doi: 10.1109/TSA.2002.800560.

[12] T. Giannakopoulos and A. Pikrakis, *Introduction to Audio Analysis: A MATLAB Approach*. Academic Press, 2014. [Online]. Available: https://www.mathworks.com/academia/books/introduction-to-audio-analysis-giannkopoulos.html

[13] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*. Prentice Hall, 2010.

[14] X. Li, K. Lee, J. Park, and S. Cho, "Audio feature extraction and classification using entropy-based methods," *IEEE Trans. Multimedia*, vol. 19, no. 2, 2017.

[15] G. Peeters, "A large set of audio features for sound description (similarity and classification) in the CUIDADO project," Paris, 2004. [Online]. Available: https://recherche.ircam.fr/anasyn/peeters/ARTICLES/Peeters_2003_cuidadoaudiofeatures.pdf

[16] D. Jiang and others, "Spectral entropy for robust speech recognition," *ICASSP*, 2011.

[17] J. Foote, "Automatic Audio Segmentation using a Measure of Audio Novelty," in *Proc. IEEE Int. Conf. on Multimedia and Expo (ICME)*, New York, USA, 2000. doi: 10.1109/ICME.2000.869637.

[18] S. Davis and P. Mermelstein, "Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences," *IEEE Trans. Acoust*, vol. 28, no. 4, 1980, doi: 10.1109/TASSP.1980.1163420.

[19] M. Slaney, "Auditory Toolbox Version 2," 1998. [Online]. Available: https://engineering.purdue.edu/~malcolm/interval/1998-010/AuditoryToolboxTechReport.pdf

[20] B. Logan, "Mel Frequency Cepstral Coefficients for Music Modeling," in *Proceedings of the 1st International Symposium on Music Information Retrieval (ISMIR)*, Plymouth, MA, USA, 2000. [Online]. Available: https://ismir2000.ismir.net/papers/logan_abs.pdf

[21] V. Kulyukin, S. Mukherjee, and P. Amlathe, "Toward Audio Beehive Monitoring: Deep Learning vs. Standard Machine Learning in Classifying Beehive Audio Samples," *Applied Sciences*, vol. 8, no. 9, p. 1573, Sep. 2018, doi: 10.3390/app8091573.

[22] J. A. Calvo, "Open Source Beehives Project - Audio Database as of 2-21-17," 2017. doi: 10.5281/zenodo.321345.

[23] icarodelimarodrigues, "Queen Loss Event in Africanized Honeybee Colony." [Online]. Available: https://www.kaggle.com/datasets/icarodelimarodrigues/queen-loss-event-in-africanized-honeybee-colony

[24] yevheniiklymenko, "Beehive buzz anomalies," 2021. [Online]. Available: https://www.kaggle.com/datasets/yevheniiklymenko/beehive-buzz-anomalies

[25] C. Montaño, "Bumblebee Repository," 2026. [Online]. Available: https://github.com/carlos-montano-hub/bumblebee

[26] A. Zgank, «Bee Swarm Activity Acoustic Classification for an IoT-Based Farm Service,» Sensors, vol. 20, dic. de 2019. DOI: 10.3390/s20010021.

Tabla de roles de contribución

Rol	Autor (es)
Conceptualización	Carlos Humberto Montaño Alcalá
Curación de datos	Carlos Humberto Montaño Alcalá
Metodología	María Trinidad Serna Encinas
Administración del Proyecto	Carlos Humberto Montaño Alcalá (igual) - María Trinidad Serna Encinas (igual).
Recursos	Carlos Humberto Montaño Alcalá (igual) - Fredy Alberto Hernández Aguirre (igual).
Software	Carlos Humberto Montaño Alcalá
Supervisión	María Trinidad Serna Encinas
Validación	Fredy Alberto Hernández Aguirre
Visualización	Carlos Humberto Montaño Alcalá (igual) - María Trinidad Serna Encinas (igual).
Redacción, borrador original.	Carlos Humberto Montaño Alcalá (igual) - María Trinidad Serna Encinas (igual).
Redacción, revisión y edición.	Carlos Humberto Montaño Alcalá (igual) - María Trinidad Serna Encinas (igual) - Fredy Alberto Hernández Aguirre (igual).