FACULTAD 50 AÑOS DE INGENIERÍA UNaM
2024 JIDeTEV ISSN 2591-4219
XIV Jornadas de Investigación, Desarrollo Tecnológico, Extensión, Vinculación y Muestra de la Producción

Machine Learning en la Apicultura: Clasificación Avanzada del Estado de Colmenas con *Pipeline* Optimizado y Etiquetado Preciso

Axel A. Skrauba [a, b *], Hector De Sosa [a], Sandro D. Zakowicz [a]

[a] *Universidad Nacional de Misiones, Facultad de Ingeniería, Oberá, Misiones, Argentina.*
[b] *GIDE, IMAM UNaM-CONICET, FI-UNaM, Oberá, Misiones, Argentina.*
e-mails: axel.skrauba@fio.unam.edu.ar, hectordesosa.01@gmail.com, sandrodanielzakowicz@gmail.com

Resumen

En este estudio, se explora un método no invasivo basado en inteligencia artificial y procesamiento de audio para la clasificación de estados en colmenas de abejas (*Apis Mellifera*). Identificar con precisión estados como la presencia o ausencia de la reina, la aceptación o rechazo de una nueva reina y otras condiciones, son cruciales para mantener la salud y productividad de las colmenas, por lo que, resultan en aspectos vitales en la apicultura. Tradicionalmente, la verificación de estos estados requiere intervención manual, lo que puede perturbar a las abejas y afectar negativamente a la colonia. Este trabajo propone la utilización de coeficientes cepstrales del espectrograma de Mel y técnicas de *machine learning*, optimizadas mediante el *framework* PyCaret, para clasificar estos estados a partir del análisis del zumbido de las colmenas. Dado que un porcentaje de los datos de audio está etiquetado incorrectamente, se ha implementado Cleanlab para corregir estas etiquetas y mejorar la precisión del modelo. El artículo no solo aborda la clasificación precisa de estos estados, sino que también realiza un análisis del impacto de la corrección de etiquetas en el rendimiento del modelo. Los resultados muestran que este enfoque tiene el potencial de ser implementado en una aplicación móvil, proporcionando una herramienta práctica para los apicultores en la gestión de sus colmenas.

Palabras Clave *– Abejas, Apicultura, Corrección de Etiquetas, Clasificación de Sonido, Extracción de Características, Machine Learning, Monitoreo de Colmenas, Preprocesamiento.*

1 Introducción

1.1 Contexto y Motivación

La reina es fundamental para la supervivencia y productividad de una colmena, ya que es la única abeja que puede poner huevos fertilizados, lo que garantiza la continuidad de la colonia. La presencia o ausencia de la reina puede indicar el estado de salud de la colmena. Si la reina está ausente o no es aceptada, la productividad y estabilidad de la colmena se ven comprometidas.

Tradicionalmente, los apicultores inspeccionan manualmente las colmenas para verificar la presencia de la reina y el estado general de la colonia. Sin embargo, este proceso es invasivo y puede perturbar a las abejas, lo que podría llevar a un estrés innecesario en la colonia y afectar negativamente la producción de miel [1].

Para evitar la intrusión directa en las colmenas, se propone un enfoque no invasivo que utiliza el análisis del sonido del zumbido de las abejas como un indicador del estado de la colmena. Los cambios en los patrones de zumbido pueden revelar la presencia o ausencia de la reina, así como otros estados importantes de la colmena [2], [3].

El desarrollo de un modelo de *machine learning* que pueda clasificar estos estados basados en el sonido abre la posibilidad de crear una aplicación móvil. Esta herramienta proporcionaría a los apicultores una forma sencilla y no invasiva de monitorizar la salud de sus colmenas, mejorando la gestión apícola.

* axel.skrauba@fio.unam.edu.ar

1.2 *Objetivos del Estudio*

En este estudio se plantean tres objetivos principales que se centran en el desarrollo y optimización de un modelo de clasificación basado en *machine learning* para la identificación de estados en colmenas de abejas, utilizando análisis de audio. A continuación, se detallan los objetivos específicos:

1.2.1 Desarrollar un modelo de clasificación para determinar los estados de la colmena

El primer objetivo es diseñar y entrenar un modelo de *machine learning* capaz de clasificar los diferentes estados de una colmena en cuatro clases distintas utilizando grabaciones de audio del zumbido de las abejas. Este enfoque no invasivo tiene el potencial de revolucionar la apicultura al permitir un monitoreo continuo y en tiempo real del estado de las colmenas sin la necesidad de inspecciones manuales que podrían perturbar a las abejas. Concretamente, se abordarán distintos estados en interés en la colmena, tomando como foco a la reina.

1.2.2 Corregir etiquetas incorrectas en los datos

Uno de los desafíos inherentes al manejo de datos etiquetados manualmente es la posibilidad de errores en las etiquetas. Estos errores pueden afectar significativamente el rendimiento de los modelos de *machine learning*, ya que introducen ruido en el proceso de entrenamiento. Cleanlab [4], una herramienta reciente y emergente en el campo de la corrección de etiquetas, se utilizará en este estudio para identificar y corregir posibles etiquetas erróneas en el conjunto de datos de audio. Este *framework*, implementa Confident Learning, en dónde se utiliza un modelo predictivo base para estimar las probabilidades de pertenencia de cada instancia respecto a las clases objetivo.

1.2.3 Evaluar el impacto de la corrección de etiquetas y optimización del pipeline en el rendimiento del modelo

Finalmente, el estudio también se enfoca en evaluar cómo la corrección de etiquetas y la optimización del *pipeline*, mediante el uso del *framework* PyCaret [5], impactan el rendimiento general del modelo de clasificación. La optimización del *pipeline* es esencial para asegurar que el modelo alcance su máximo potencial en términos de precisión y generalización. Se analizarán diferentes combinaciones de parámetros y arquitecturas de modelos para determinar la configuración que ofrezca los mejores resultados. Además, se realizará un análisis comparativo del rendimiento del modelo antes y después de la aplicación de Cleanlab, lo que permitirá medir el valor añadido de la corrección de etiquetas en este contexto.

2 Revisión de Literatura

2.1 *Procesamiento de Audio en Apicultura*

El uso del sonido como herramienta para el estudio de las colmenas de abejas no es una práctica nueva, pero ha ganado relevancia significativa en las últimas décadas debido a los avances en tecnología de grabación y procesamiento de señales. Los primeros estudios se centraron en la observación del comportamiento de las abejas mediante grabaciones de audio rudimentarias, buscando correlaciones entre ciertos sonidos y el estado de la colmena. Estos métodos permitieron a los investigadores identificar patrones de zumbido que indicaban la presencia de la reina, enjambres inminentes, o estados de estrés dentro de la colonia [6].

A medida que la tecnología avanzó, también lo hicieron las aplicaciones del análisis de sonido en la apicultura. Hoy en día, se emplean técnicas sofisticadas de procesamiento de señales y *machine learning* para interpretar los datos de audio de manera más precisa y automatizada [3]. No obstante, a pesar de los avances, los métodos actuales todavía enfrentan limitaciones significativas, como la variabilidad en la calidad de las grabaciones y la necesidad de un etiquetado manual preciso de los datos, lo que introduce un nivel considerable de error humano en el proceso.

2.2 Extracción de Características en Audio

En el campo del procesamiento de audio, la extracción de características es una etapa crucial que determina la efectividad del modelo de clasificación. Existen varios métodos para extraer características relevantes de señales de audio, cada uno con sus propias ventajas y desventajas. Entre los métodos más comunes se encuentran los coeficientes cepstrales en las frecuencias de Mel (MFCC), la transformada rápida de Fourier (FFT), y más recientemente, técnicas especializadas en la captura de descriptores en general, que se sustentan en los *embeddings* de modelos profundos, entrenados para tareas generales. En el caso concreto de audio, un ejemplo de esto sería YAMNet [7].

Los MFCCs han sido ampliamente adoptados en aplicaciones de reconocimiento de voz y se han adaptado con éxito al análisis de audio en apicultura debido a su capacidad para capturar la envolvente espectral de las señales de zumbido, que es fundamental para la diferenciación de estados en la colmena [3]. Por otro lado, la FFT proporciona una representación frecuencial directa de la señal, lo que puede ser útil para identificar patrones específicos en el zumbido de las abejas. Sin embargo, los MFCCs son preferidos en este estudio debido a su eficacia en el modelado de señales complejas como las generadas por las abejas, lo que justifica su selección como método principal para la extracción de características en este trabajo.

2.3 Optimización de Pipelines

La optimización de *pipelines* es un componente esencial en cualquier proyecto de inteligencia artificial, ya que permite maximizar el rendimiento del modelo mediante la selección adecuada de hiperparámetros y arquitecturas, además de factores involucrados en el preprocesamiento de los descriptores. Herramientas como PyCaret han facilitado este proceso al proporcionar una plataforma que automatiza la selección de modelos, la optimización de hiperparámetros, y la evaluación comparativa, lo que ahorra tiempo y recursos a los investigadores.

En el contexto de este estudio, PyCaret se utilizará para construir y optimizar el *pipeline* de clasificación. Estudios previos han demostrado que la optimización cuidadosa del *pipeline* puede llevar a mejoras significativas en el rendimiento de los modelos, especialmente cuando se combinan con técnicas de corrección de etiquetas como Cleanlab [4]. Estas mejoras son cruciales en aplicaciones como la apicultura, donde la precisión del modelo puede tener un impacto directo en la salud y productividad de las colmenas, y, por ende, en los ingresos de los apicultores.

2.4 Corrección de Etiquetas en Machine Learning

El problema de las etiquetas incorrectas en *datasets* es un desafío bien documentado en *machine learning*. Las etiquetas incorrectas pueden degradar severamente el rendimiento de los modelos,

especialmente en aplicaciones donde la precisión es crítica. La corrección de etiquetas ha sido un área de estudio creciente, con varias metodologías propuestas para mitigar este problema. Entre estas, Cleanlab ha emergido como una herramienta potente que no solo detecta etiquetas incorrectas, sino que también sugiere correcciones basadas en la probabilidad de las clases predichas [4].

El uso de Cleanlab es particularmente relevante en este estudio debido a la naturaleza de los datos de audio, donde el etiquetado manual es propenso a errores. Dado que la mayoría de los *datasets* utilizados en estudios recientes son privados [8], [9] y solo unos pocos han sido liberados públicamente [10], [11], [12], la calidad de las etiquetas es una preocupación constante, ya que depende de los propósitos de cada autor y las técnicas empleadas. La implementación de Cleanlab permite mejorar la precisión del modelo al corregir estas etiquetas erróneas, lo que maximiza el valor de los datos disponibles y mejora los resultados de la clasificación.

3 Metodología

3.1 Conjunto de Datos

El conjunto de datos utilizado en este estudio proviene del *Smart Bee Colony Monitor Dataset*, disponible públicamente en la plataforma Kaggle [10]. Este *dataset* fue recopilado con el objetivo de fomentar el desarrollo de métodos computacionales que contribuyan a la protección de las abejas y estado de salud de las colmenas, mediante la detección remota y en tiempo real, través del análisis de audio y otras características intrínsecas. Las grabaciones de audio, que forman el núcleo de este estudio, fueron obtenidas de colmenas de abejas europeas (*Apis Mellifera*) ubicadas en California, utilizando un dispositivo personalizado.

El *dataset* contiene un total de 7100 muestras de audio, cada una de las cuales es un segmento de 60 segundos. Estas muestras derivan de un conjunto original de poco más de 1200 grabaciones largas, que fueron recortadas en los segmentos de 60 segundos para su análisis. Cada archivo de audio se asocia con una serie de metadatos que incluyen la fecha, hora, ubicación, presencia de la reina, aceptación de la reina, temperatura, humedad y condiciones climáticas. Sin embargo, para este estudio, se utilizarán solo las grabaciones de audio, prescindiendo de los datos contextuales adicionales, ya que el objetivo es desarrollar un modelo que pueda ser eventualmente implementado en una aplicación móvil, utilizando únicamente el micrófono del dispositivo como entrada.

El enfoque principal del estudio es la clasificación de los estados de la reina en la colmena, a partir de los zumbidos del enjambre.

Los estados considerados son cuatro y se catalogan como:

- 0 - reina original presente
- 1 - reina no presente
- 2 - reina presente pero rechazada
- 3 - reina presente y recién aceptada

En la Fig. 1, puede apreciarse el espectrograma de una muestra de audio para cada una de estas clases. Denotando frecuencias y patrones para cada estado de interés.

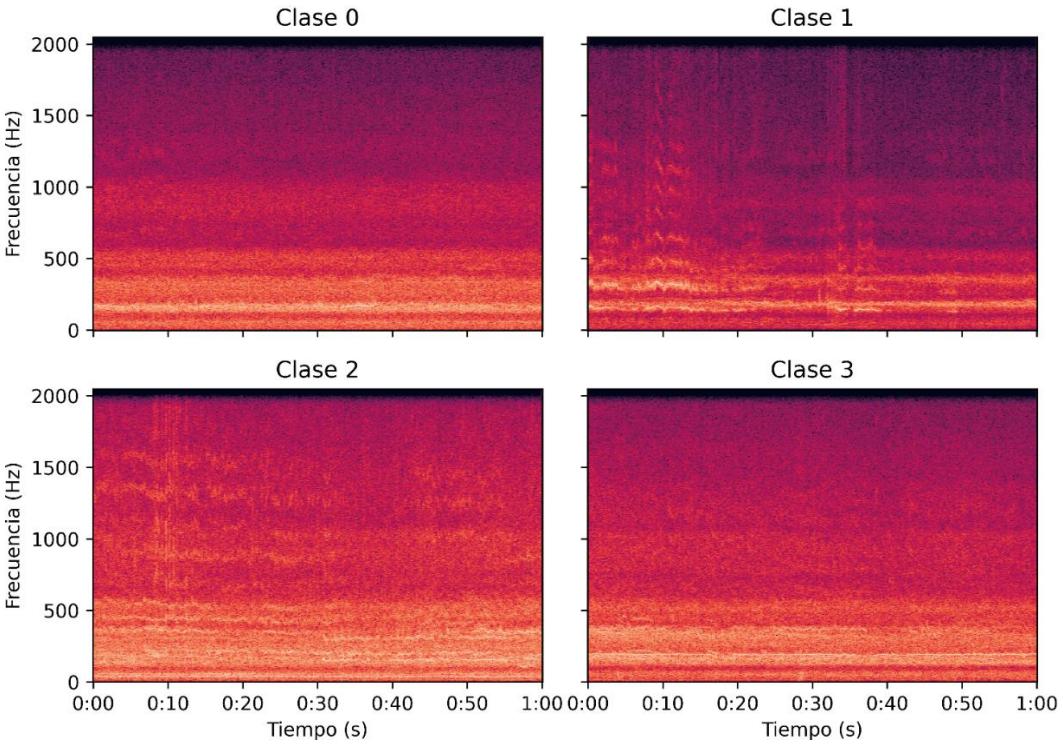

Fig. 1. Espectrogramas de los 4 estados/clases considerados.

La clasificación precisa entre estos estados es crucial para la gestión eficiente de la colmena, ya que la presencia y aceptación de la reina son indicadores fundamentales de la salud y estabilidad de la colonia. En la actualidad, la mayoría de los estudios se centra en determinar si la reina está o no en la colmena, es decir, proponen un clasificador binario [8], [13], [14]. El objetivo es poder discriminar estados más específicos, ya que la reina puede estar, pero la colonia no la acepta (caso crítico al implantar una nueva, por ejemplo). Identificar este tipo de cuestiones, sería clave para los apicultores, de modo de poder realizar las maniobras adecuadas en sus colmenas.

Una consideración importante al utilizar este conjunto de datos es el hecho de que el etiquetado de los estados de la reina fue realizado manualmente por los autores del *dataset* antes del proceso de segmentación. Esto implica que cualquier error humano cometido durante el etiquetado original se propaga a todos los segmentos derivados de esa muestra. Este es un aspecto crítico, dado el tamaño limitado del *dataset*, y subraya la importancia de utilizar técnicas avanzadas de corrección de etiquetas, para maximizar la utilidad de los datos disponibles y mejorar la precisión del modelo.

Para asegurar una evaluación robusta del modelo, se tomará en cuenta la procedencia de los fragmentos de audio al momento de dividir los datos en conjuntos de entrenamiento y prueba (70/30). Específicamente, se evitará que fragmentos derivados de la misma muestra original se crucen entre estos conjuntos, minimizando así la posibilidad de *data leakage* y asegurando que la validación del modelo refleje de manera precisa su capacidad de generalización.

3.2 Extracción de Características

Para el análisis de los sonidos de las colmenas, se remuestrearon los audios utilizando una frecuencia de 4 kHz, seleccionada en base a estudios previos que indican que las frecuencias relevantes para la detección de estados en colmenas de abejas se encuentran típicamente en el rango

de 100 Hz a 2000 Hz [3]. Esta frecuencia de remuestreo permite capturar adecuadamente las variaciones espectrales significativas sin introducir ruido de alta frecuencia, no relevante para la clasificación de los estados de la colmena y reduciendo la carga de los modelos a implementar.

Una vez remuestreados los audios, se procedió a la extracción de características utilizando los MFCCs. Para cada segmento de audio, se calcularon 50 coeficientes MFCC, utilizando un ventaneo estándar proporcionado por la biblioteca Librosa.

Los MFCCs se calcularon para ventanas de tiempo acotadas dentro de cada segmento de audio, y posteriormente, se tomó la media de cada coeficiente a lo largo del segmento. Esta aproximación produce un vector de 50 características por segmento de audio, representando la información espectral promedio del mismo. La media de coeficientes en el tiempo responde a la naturaleza del problema; se espera que las variaciones en los zumbidos no presenten cambios rápidos, lo que hace viable la simplificación del análisis espectral en ventanas temporales acotadas y mitiga efectos ambientales como impactos externos sobre la colmena.

Este enfoque, centrado en la extracción de MFCCs, resultó ser significativamente más efectivo en comparación con otros métodos de extracción de características, como los *embeddings* generados por YAMNet. La simplicidad y eficiencia computacional de los MFCCs, combinada con su efectividad en la tarea, justifican su selección como la característica principal para el modelo de clasificación de estados de la colmena.

3.3 Optimización del Pipeline de Clasificación

Para optimizar el *pipeline* de clasificación y asegurar que los modelos empleados ofrecieran un rendimiento óptimo, se evaluaron inicialmente los modelos base de PyCaret, incluyendo Logistic Regression, K Neighbors Classifier, Naive Bayes, Decision Tree Classifier, Support Vector Machines (con *kernel* lineal y radial), Random Forest, AdaBoost, Gradient Boosting Classifier, y Extra Trees Classifier, entre otros (Tabla 1).

PyCaret permitió evaluar el desempeño de cada modelo utilizando un conjunto estándar de métricas: *Accuracy*, AUC, *Recall*, *Precision*, *F1-Score*, MCC (*Matthews Correlation Coefficient*), y el tiempo de entrenamiento (TT, *Time to Train*). Estas métricas ofrecieron una visión integral del rendimiento de cada modelo, permitiendo la identificación de los más prometedores para la clasificación de los estados de la colmena.

El proceso de optimización no se limitó a los hiperparámetros de los modelos individuales, sino que se centró en la optimización global del *pipeline* de clasificación. Se llevaron a cabo experimentos que incluían la normalización de características, la aplicación de PCA (Análisis de Componentes Principales), la remoción de *outliers*, y el balanceo de clases. Cada uno de estos enfoques fue evaluado en términos de su impacto en el rendimiento global, utilizando la validación cruzada con 10 pliegues (10-*fold cross-validation*) para asegurar que los resultados fueran representativos y generalizables.

3.4 Corrección de Etiquetas con Cleanlab

En nuestra metodología, Cleanlab se integró durante la fase del análisis exploratorio de datos. En primer lugar, se extrajeron 50 coeficientes MFCC de cada segmento de audio, que fueron utilizados como *features* para alimentar un modelo de Random Forest, utilizado como base en la

implementación de Confident Learning. A través de una validación cruzada de 15 *folds*, se generaron las probabilidades de predicción para cada segmento, las cuales fueron luego analizadas por Cleanlab para identificar posibles problemas de etiquetado.

El proceso de corrección se realizó en dos niveles:

- **A nivel de segmento individual**: Se analizaron los segmentos aislados para determinar si presentaban inconsistencias que pudieran indicar errores de etiquetado debido al procesamiento o al corte de los segmentos (nativo de los autores del *dataset*).
- **A nivel de audio original**: Dado que cada audio original se segmentó en 6 partes, se implementó un criterio de corrección basado en la coincidencia de etiquetas. Si al menos tres segmentos coincidían en otra categoría diferente a la originalmente etiquetada, se consideró que el audio original había sido etiquetado erróneamente, y se procedió a corregir la etiqueta para todos sus segmentos.

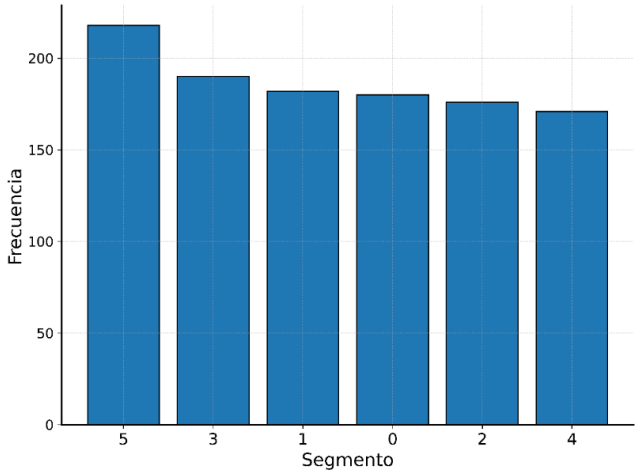

Fig. 2. Distribución de inconsistencias en los segmentos individuales.

En la Fig. 2, se observa la distribución de posibles errores de etiquetas, discriminadas por segmentos. Es de esperar, que la misma cuente con una distribución uniforme, ya que los segmentos surgen de un mismo audio de origen. Sin embargo, se denota una frecuencia superior para los segmentos número 5. En un esfuerzo por intentar entender el fenómeno, se ejemplifica en la Fig. 3, la FFT para cada uno de los segmentos de un mismo audio. Debieran ser parecidos, ya que corresponden al mismo evento dentro de la colmena, pero se denota una diferencia sustancial para el segmento 5.

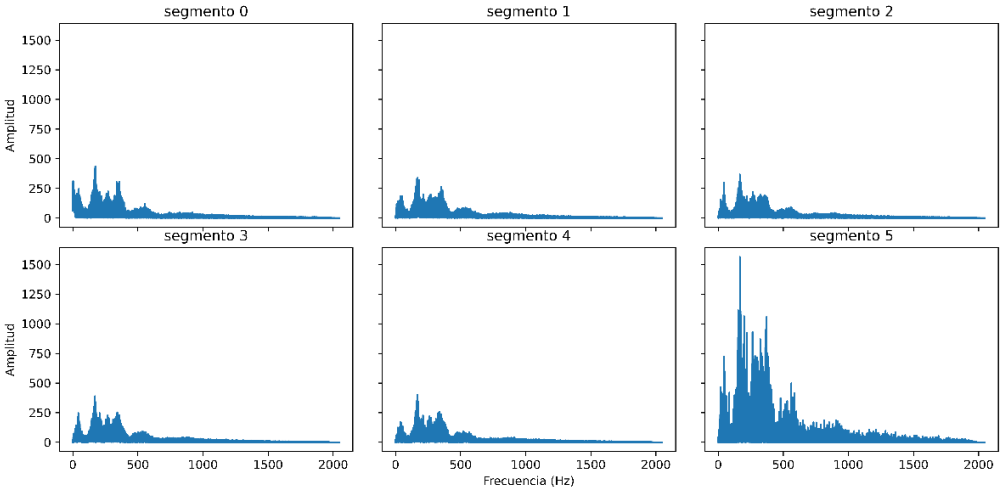

Fig. 3. FFT para los segmentos sucesivos de un mismo audio.

Este fenómeno en los segmentos 5 se observó para todos los audios, al analizarlos en detalle, es evidente que presentan efecto de *aliasing*. Se desconoce el origen de este problema debido a que no se observa en otros segmentos y proviene directamente del *dataset* original. En la Fig. 4 se observan varios ejemplos del fragmento 5 para múltiples audios y múltiples clases, todos tienen el problema de *aliasing* que se observa. Por lo tanto, se descartaron para el entrenamiento y la validación de los modelos.

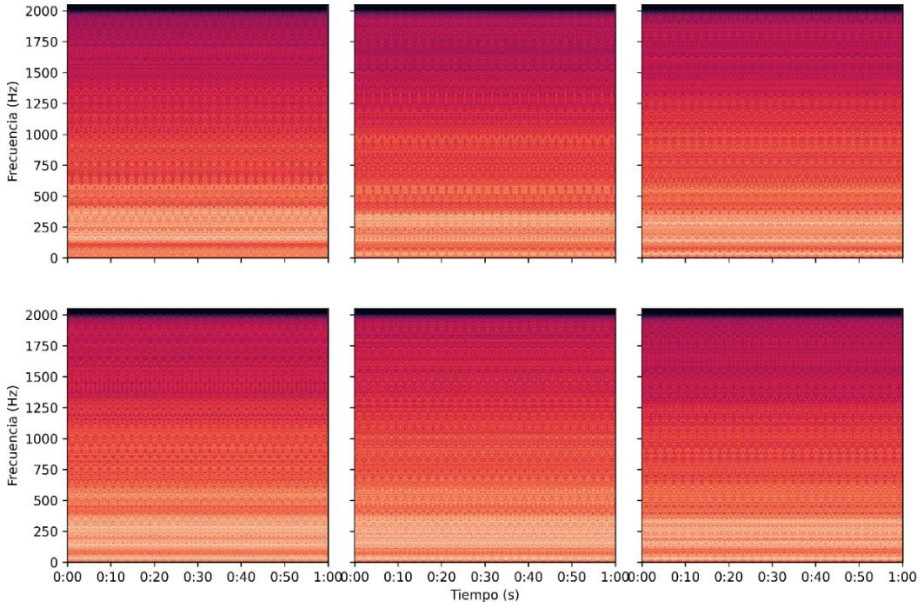

Fig. 4. Ejemplos del espectro para el segmento 5, múltiples audios y múltiples estados de la reina.

Tras aplicar Cleanlab con el criterio establecido, se redefinieron alrededor del 10% de las etiquetas de los segmentos originales. La mejora en el rendimiento de los modelos se analizará en detalle en la sección de resultados, confirmando la existencia de errores en el etiquetado inicial mediante la comprobación empírica entre el desempeño de los modelos en ambos casos (etiquetas originales y corregidas).

En la Fig. 5 se muestran dos casos de corrección de etiquetas. Las figuras de la izquierda (primera columna) se tomaron como referencia (alta confianza según Cleanlab) para las clases 0 y 2. En las

figuras de la derecha (segunda columna) se observan espectrogramas cuyas etiquetas fueron reclasificadas, pudiendo visualizar coincidencias entre las distintas bandas en frecuencias respecto a la supuesta clase original y la corregida. En los ejemplos reclasificados, la leyenda "Clase 2 a 0" significa que; originalmente la muestra era de clase 2 pero se reclasificó como de clase 0, ídem para el otro caso.

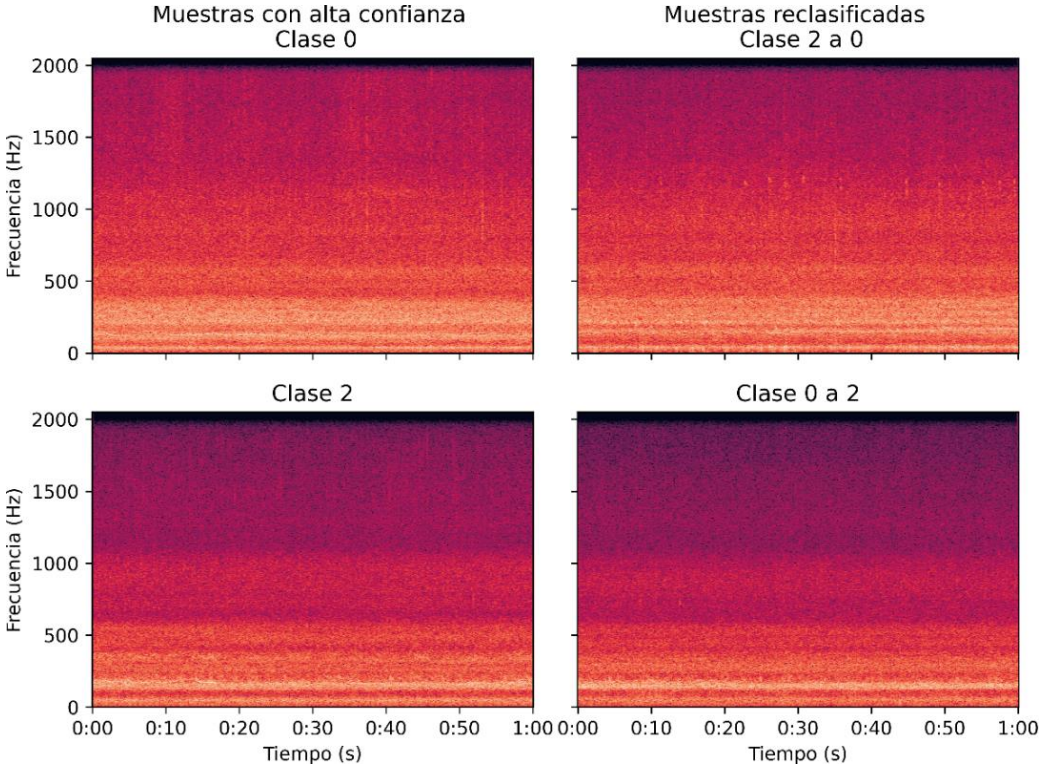

Fig. 5. Ejemplos del espectro para audios con etiquetas corregidas.

4 Resultados y Discusión

4.1 Resultados de la Clasificación Inicial

En esta sección, se presentan los resultados obtenidos al aplicar una serie de modelos de clasificación a los datos iniciales, utilizando como características los descriptores MFCC extraídos previamente. Es importante destacar que estos resultados se obtuvieron antes de realizar cualquier optimización del *pipeline* o corrección de etiquetas, lo que nos permitirá establecer una línea base para contrastar con los resultados mejorados en etapas posteriores.

Para la evaluación inicial, se emplearon todos los modelos de clasificación rápidos disponibles en PyCaret, lo que incluye una selección estándar de algoritmos supervisados de clasificación. Los modelos se entrenaron y evaluaron utilizando los valores crudos de los 50 coeficientes MFCC, sin aplicar técnicas de preprocesamiento.

La Tabla 1 resume las métricas clave obtenidas para cada modelo:

Tabla 1 – Desempeño base de los modelos, sin optimización del *pipeline*

Modelo/Métrica	***Accuracy***	***AUC***	***Recall***	***Prec.***	***F1***	***MCC***	**TT (S)**
Extra Trees Classifier	**0.6296**	0.8198	**0.6296**	**0.6369**	0.6005	0.4210	0.0340
CatBoost Classifier	0.6287	**0.8231**	0.6287	0.6340	**0.6053**	**0.4270**	0.7090

Random Forest Classifier	0.6231	0.8142	0.6231	0.6264	0.5946	0.4112	0.0490
Light Gradient Boosting Machine	0.6212	0.8143	0.6212	0.6246	0.5971	0.4151	0.3360
Extreme Gradient Boosting	0.6186	0.8108	0.6186	0.6263	0.5970	0.4113	0.0520
Gradient Boosting Classifier	0.6071	0.0000	0.6071	0.5851	0.5598	0.3720	0.2770
K Neighbors Classifier	0.6052	0.7732	0.6052	0.6202	0.5868	0.3978	0.0130
Ridge Classifier	0.5977	0.0000	0.5977	0.4486	0.4836	0.3076	0.0060
Linear Discriminant Analysis	0.5965	0.0000	0.5965	0.4619	0.4876	0.3149	0.0050
Logistic Regression	0.5960	0.0000	0.5960	0.4663	0.4882	0.3104	0.0110
Naive Bayes	0.5836	0.7139	0.5836	0.5134	0.5013	0.3131	0.0060
Quadratic Discriminant Analysis	0.5833	0.0000	0.5833	0.5216	0.5138	0.3178	0.0060
Decision Tree Classifier	0.5557	0.6776	0.5557	0.5786	0.5502	0.3363	0.0080
Ada Boost Classifier	0.5524	0.0000	0.5524	0.5188	0.4996	0.2725	0.0270
Dummy Classifier	0.5100	0.5000	0.5100	0.2601	0.3445	0.0000	0.0060
SVM - Linear Kernel	0.4533	0.0000	0.4533	0.4294	0.4153	0.1510	0.0130

En esta evaluación inicial, el Extra Trees Classifier mostró el mejor rendimiento con una *Accuracy* de 0,6296 y un AUC de 0,8198. Le siguieron de cerca el CatBoost Classifier y el Random Forest Classifier con *Accuracy* de 0,6287 y 0,6231, respectivamente. Estos resultados reflejan un rendimiento modesto, lo que sugiere que el *pipeline* y los datos requieren optimización para mejorar la capacidad predictiva de los modelos, que los *features* utilizados no presentan la información necesaria o que el problema es difícil de resolver para este tipo de modelos.

Por otro lado, modelos como SVM - Linear Kernel y Dummy Classifier obtuvieron los peores resultados, con *Accuracy* de 0,4533 y 0,5100, respectivamente. Estos valores refuerzan la idea de mejorar el *pipeline* para obtener resultados más robustos.

Estos resultados iniciales proporcionan una línea base esencial, que nos permitirá evaluar de manera efectiva el impacto de las optimizaciones del *pipeline* y la corrección de etiquetas, las cuales serán discutidas en las siguientes subsecciones.

4.2 Optimización del Pipeline

Para mejorar los resultados obtenidos en la clasificación inicial, se procedió a optimizar el *pipeline* de procesamiento de los datos. La mejor configuración identificada consistió en tres pasos clave:

- **Remoción de *Outliers***: Se empleó el algoritmo de Isolation Forest con un umbral de contaminación del 5% para eliminar muestras atípicas que podrían estar afectando negativamente la precisión del modelo. La remoción de estos *outliers* resultó en una mejora notable en la precisión global, dado que permitió que los modelos se concentraran en patrones más representativos de las clases. En particular, se observó un aumento en la precisión de los modelos más complejos, como el Random Forest y el CatBoost (Tabla 2), lo que subraya la importancia de eliminar datos ruidosos (se encontraron muestras con superposición de sonidos de maquinaria agrícola, por dar un ejemplo).
- **Balanceo de Clases**: Dado que las clases no estaban equilibradas, se utilizó la técnica SMOTE (*Synthetic Minority Over-sampling Technique*) con 5 vecinos cercanos y estrategia de muestreo automática. Este paso fue crucial para mejorar el *Recall*, especialmente en las clases minoritarias.

Al generar muestras sintéticas de estas clases, se evitó que los modelos se sesgaran hacia las clases más representadas, lo que permitió una mejor generalización y un incremento en el rendimiento de modelos como el CatBoost y el Gradient Boosting, donde el *Recall* mostró una mejora significativa (Tabla 2).

- **Normalización**: Finalmente, se aplicó la normalización de los datos mediante *Standard Scaler*, que estandarizó los descriptores MFCC, asegurando que cada uno tuviera una media de 0 y una desviación estándar de 1. Esta normalización es particularmente beneficiosa para modelos que dependen de la medición de distancias entre puntos en el espacio de características, como K Neighbors Classifier (KNN), de modo de permitir explorar soluciones en igualdad de condiciones respecto a modelos no sensibles (basados en árboles, por ejemplo).

El efecto de esta etapa se observa en la Tabla 2:

Tabla 2 – Desempeño de los modelos, con optimización del *pipeline*

Modelo/Métrica	*Accuracy*	*AUC*	*Recall*	*Prec.*	*F1*	*MCC*	TT (S)
CatBoost Classifier	**0.8222**	**0.9634**	**0.8222**	**0.8442**	**0.8196**	**0.7431**	4.3480
Extreme Gradient Boosting	0.8042	0.9573	0.8042	0.8249	0.7997	0.7143	0.3100
Quadratic Discriminant Analysis	0.8038	0.0000	0.8038	0.8297	0.7916	0.7119	0.0220
Light Gradient Boosting Machine	0.7963	0.9534	0.7963	0.8157	0.7905	0.7010	0.5480
Random Forest Classifier	0.7884	0.9498	0.7884	0.8104	0.7785	0.6872	0.2180
Extra Trees Classifier	0.7819	0.9513	0.7819	0.8046	0.7682	0.6766	0.0690
Gradient Boosting Classifier	0.7515	0.0000	0.7515	0.7750	0.7475	0.6365	5.1470
K Neighbors Classifier	0.7474	0.8970	0.7474	0.8015	0.7474	0.6525	0.1630
Logistic Regression	0.7428	0.0000	0.7428	0.7838	0.7439	0.6352	0.2210
SVM - Linear Kernel	0.7376	0.0000	0.7376	0.7709	0.7382	0.6206	0.0300
Linear Discriminant Analysis	0.7337	0.0000	0.7337	0.7754	0.7357	0.6236	0.0200
Ridge Classifier	0.7280	0.0000	0.7280	0.7623	0.7279	0.6096	0.0210
Decision Tree Classifier	0.6390	0.7486	0.6390	0.6769	0.6388	0.4765	0.0610
Naive Bayes	0.5922	0.7805	0.5922	0.6082	0.5672	0.3740	0.0200
Ada Boost Classifier	0.5862	0.0000	0.5862	0.6598	0.5973	0.4171	0.2400
Dummy Classifier	0.1338	0.5000	0.1338	0.0179	0.0316	0.0000	0.0240

Tras implementar este *pipeline*, los resultados mostraron mejoras significativas en comparación con la clasificación inicial (Tabla 1). Por ejemplo, el CatBoost Classifier, que ya era uno de los mejores en el *baseline*, mejoró su *Accuracy* de 0,6287 a 0,8222 y su *F1-Score* de 0,6053 a 0,8196. De igual manera, el Random Forest Classifier experimentó un aumento en su *Accuracy* de 0,6231 a 0,7884, demostrando que la optimización del *pipeline* permitió a los modelos capturar patrones más significativos en los datos.

4.3 Impacto de la Corrección de Etiquetas

La corrección de etiquetas llevada a cabo, tuvo un impacto significativo en el rendimiento de los modelos evaluados. Al aplicar el *pipeline* optimizado que incluye la remoción de *outliers*, el balanceo

de clases, y la normalización de los descriptores MFCC; se observaron mejoras notables en las diversas métricas de clasificación. Resultados a continuación:

Tabla 3 – Desempeño de los modelos, con optimización del *pipeline* y corrección de etiquetas

Modelo/Métrica	***Accuracy***	***AUC***	***Recall***	***Prec.***	***F1***	***MCC***	**TT (S)**
CatBoost Classifier	**0.8469**	**0.9704**	**0.8469**	**0.8617**	**0.8418**	**0.7768**	4.3150
Light Gradient Boosting Machine	0.8400	0.9662	0.8400	0.8529	0.8332	0.7640	0.5430
Extreme Gradient Boosting	0.8364	0.9677	0.8364	0.8510	0.8301	0.7593	0.2660
Random Forest Classifier	0.8232	0.9650	0.8232	0.8401	0.8125	0.7387	0.2120
Extra Trees Classifier	0.8131	0.9649	0.8131	0.8285	0.7976	0.7229	0.0670
Quadratic Discriminant Analysis	0.8073	0.0000	0.8073	0.8383	0.7882	0.7162	0.0230
Gradient Boosting Classifier	0.8069	0.0000	0.8069	0.8254	0.8021	0.7175	4.9320
SVM - Linear Kernel	0.7910	0.0000	0.7910	0.8139	0.7872	0.6975	0.0320
Logistic Regression	0.7843	0.0000	0.7843	0.8123	0.7836	0.6893	0.0350
K Neighbors Classifier	0.7803	0.9078	0.7803	0.8261	0.7837	0.6923	0.0280
Ridge Classifier	0.7663	0.0000	0.7663	0.7960	0.7615	0.6640	0.0210
Linear Discriminant Analysis	0.7661	0.0000	0.7661	0.8051	0.7639	0.6685	0.0220
Decision Tree Classifier	0.6982	0.7869	0.6982	0.7249	0.6966	0.5592	0.0630
Naive Bayes	0.6618	0.8290	0.6618	0.6716	0.6463	0.4872	0.0200
Ada Boost Classifier	0.6524	0.0000	0.6524	0.7151	0.6635	0.5085	0.2300
Dummy Classifier	0.1207	0.5000	0.1207	0.0146	0.0260	0.0000	0.0240

Comparando los resultados obtenidos antes (Tabla 2) y después de la corrección de etiquetas (Tabla 3), se evidencia un incremento generalizado en todas las métricas y para la mayoría de los modelos. Por ejemplo, el CatBoost Classifier, que ya mostraba un buen rendimiento en la evaluación previa, mejoró aún más su *Accuracy* de 0,8222 a 0,8469 y su MCC de 0,7431 a 0,7768. De igual manera, modelos como Light Gradient Boosting Machine y Extreme Gradient Boosting mostraron incrementos en el AUC y *F1-Score*, reflejando un mejor equilibrio entre la precisión y el *Recall* tras la corrección de etiquetas.

El impacto específico de la corrección de etiquetas fue también notable en modelos lineales y basados en árboles. El Random Forest Classifier, por ejemplo, experimentó un aumento en su *Recall* y Precisión, logrando una *F1-Score* superior, lo que vuelve a reforzar la idea de que la limpieza de etiquetas contribuyó a una mejor discriminación entre las clases.

Además, considerando ya un factor no tan exhaustivo, la corrección afectó positivamente el TT en algunos casos, lo que sugiere que la eliminación de ruido en los datos puede llevar a un proceso de entrenamiento más eficiente, aunque en otros casos, como en el Gradient Boosting Classifier, el tiempo de entrenamiento aumentó ligeramente, lo que podría estar relacionado con la necesidad de ajustar mejor el modelo a los datos, ahora más precisos.

Tomando el modelo CatBoost Classifier, que es el que presenta los mejores resultados (tanto en la Tabla 2 como en la Tabla 3), se tiene que su desempeño sobre el conjunto de pruebas, para cada una de las clases, se corresponde con lo expuesto en las matrices de confusión de la Fig. 6.

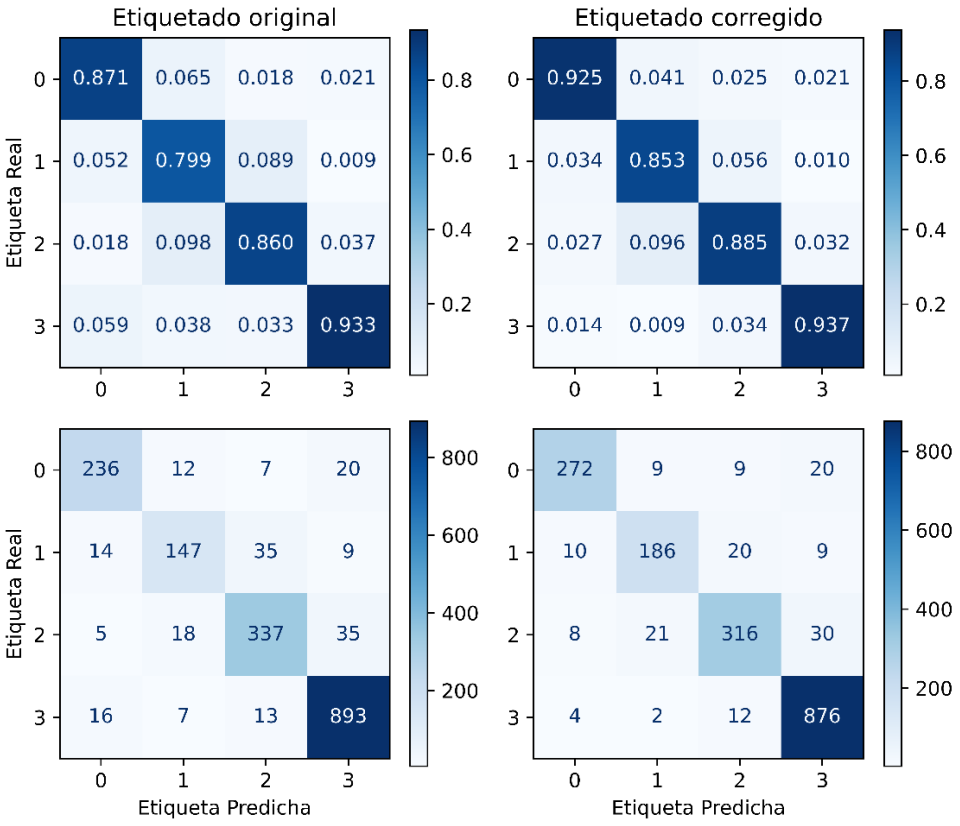

Fig. 6. Matrices de confusión sobre el set de pruebas, para etiquetas originales y etiquetas corregidas.

Las matrices de confusión de la Fig. 6 revelan que, tras la corrección de etiquetas, el modelo no solo mejora su precisión general, sino que también disminuye significativamente los errores de clasificación entre clases cercanas. Por ejemplo, el número de errores al clasificar la etiqueta 1 disminuye considerablemente, la precisión de la clase aumenta de 0,799 a 0,855 (en la primera columna de la Fig. 6 se encuentran los valores porcentuales, mientras que, en la segunda, por cantidades de observaciones predichas). En grandes rasgos, se percibe un aumento del *Accuracy* global (sobre el set de prueba) del 0,894 al 0,915. Esta mejora sugiere que las etiquetas corregidas permiten al modelo capturar patrones más consistentes y representativos de las características subyacentes, reduciendo el ruido y los errores en las predicciones.

5 Conclusiones y Trabajo Futuro

5.1 Conclusiones Principales

Este estudio ha demostrado de manera efectiva la importancia de la optimización de *pipelines* y la corrección de etiquetas en modelos de clasificación aplicados al análisis de sonidos en la apicultura. A diferencia de trabajos previos que se centraron en la clasificación binaria del estado de las colmenas de abejas (como la presencia o ausencia de la reina), este enfoque ha abordado con éxito la clasificación multiclase, lo que permite una identificación más específica y detallada de los diferentes estados de la colmena. Esta metodología, además, se ha desarrollado utilizando datos de acceso libre, lo que subraya su potencial para ser replicada y aplicada en diferentes contextos.

La implementación de un *pipeline* optimizado, que incluye la remoción de *outliers*, el balanceo de clases y la normalización de los datos, ha resultado en una mejora sustancial en las métricas de desempeño de los modelos. Específicamente, la aplicación de la corrección de etiquetas mediante Cleanlab ha permitido un incremento significativo en la precisión global del modelo, con un *Accuracy* que aumentó del 0,894 al 0,915 en el conjunto de prueba. Estos resultados subrayan la relevancia de abordar tanto la calidad de los datos como la robustez del *pipeline* en la construcción de modelos predictivos confiables para la apicultura.

El modelo desarrollado tiene un potencial significativo para ser implementado en dispositivos portátiles, ya que su capacidad para realizar inferencias no requiere un alto poder de cómputo. Esto abre la posibilidad de integrarlo en una aplicación móvil, que podría ser una herramienta invaluable para los apicultores. Dicha aplicación permitiría a los apicultores obtener información en tiempo real sobre el estado de cada una de sus colmenas, facilitando la toma de decisiones informadas y mejorando la gestión y el bienestar de las abejas.

5.2 Limitaciones del Estudio y Desafíos Futuros

A pesar de las mejoras observadas, el enfoque presenta algunas limitaciones. La principal limitación radica en la dependencia del modelo de la calidad de los datos y las etiquetas. Si bien la corrección de etiquetas ha demostrado ser efectiva en este caso, en situaciones donde las etiquetas son inherentemente ambiguas o difíciles de definir, podría ser necesario recurrir a expertos humanos para verificar o ajustar las correcciones automatizadas.

En cuanto a desafíos futuros, una limitación a considerar es la posible sobre adaptación del modelo a las características específicas del conjunto de datos utilizado. Aunque los resultados obtenidos son prometedores, la generalización del modelo a diferentes poblaciones de datos o a otras especies de abejas debe ser cuidadosamente evaluada.

El modelo actual, que ha demostrado su efectividad en la identificación de cuatro clases distintas en el análisis del sonido de colmenas, abre nuevas posibilidades para futuras investigaciones. Una de las direcciones más prometedoras sería utilizar este modelo para etiquetar automáticamente conjuntos de datos abiertos que, hasta ahora, solo han sido categorizados de manera binaria. Al aplicar esta metodología, se podría obtener información más detallada y específica sobre los estados de las colmenas en una variedad de entornos, lo que contribuiría a una comprensión más profunda y contextualizada del comportamiento de las abejas.

Además, con un volumen significativamente mayor de datos etiquetados de manera precisa, sería viable explorar la implementación de modelos basados en *deep learning*. Estos modelos, conocidos por su capacidad para captar y aprender patrones complejos, podrían mejorar aún más la precisión y la capacidad del sistema para reconocer las características distintivas de cada clase. Tal enfoque podría llevar a avances importantes en la clasificación automatizada y la monitorización en la apicultura, con aplicaciones que podrían extenderse a otras áreas del análisis de sonido en la biología.

Referencias

[1] R. A. Martínez Sarmiento, N. C. Ortega Flórez, W. D. Maldonado Quintero, y R. E. V. R. y Otros, *Manual técnico de apicultura : abeja (Apis mellifera)*. Corporación colombiana de investigación agropecuaria - AGROSAVIA, 2012. doi: 10.21930/agrosavia.manual.2012.1.

[2] C. Uthoff, M. N. Homsi, y M. von Bergen, «Acoustic and vibration monitoring of honeybee colonies for beekeeping-relevant aspects of presence of queen bee and swarming», *Comput. Electron. Agric.*, vol. 205, p. 107589, feb. 2023, doi: 10.1016/j.compag.2022.107589.
[3] M. Abdollahi, P. Giovenazzo, y T. H. Falk, «Automated Beehive Acoustics Monitoring: A Comprehensive Review of the Literature and Recommendations for Future Work», *Appl. Sci.*, vol. 12, n.º 8, Art. n.º 8, ene. 2022, doi: 10.3390/app12083920.
[4] C. G. Northcutt, L. Jiang, y I. L. Chuang, «Confident Learning: Estimating Uncertainty in Dataset Labels», 21 de agosto de 2022, *arXiv*: arXiv:1911.00068. [En línea]. Disponible en: http://arxiv.org/abs/1911.00068
[5] M. Ali, *PyCaret: An open source, low-code machine learning library in Python*. 2020. [En línea]. Disponible en: https://www.pycaret.org
[6] D. Howard, O. Duran, G. Hunter, y K. Stebel, «Signal processing the acoustics of honeybees (APIS MELLIFERA) to identify the "queenless" state in Hives», *Proc. Inst. Acoust.*, vol. 35, pp. 290-297, ene. 2013.
[7] S. Hershey *et al.*, «CNN Architectures for Large-Scale Audio Classification», 10 de enero de 2017, *arXiv*: arXiv:1609.09430. doi: 10.48550/arXiv.1609.09430.
[8] S. Ruvinga, G. Hunter, O. Duran, y J.-C. Nebel, «Identifying Queenlessness in Honeybee Hives from Audio Signals Using Machine Learning», *Electronics*, vol. 12, n.º 7, Art. n.º 7, ene. 2023, doi: 10.3390/electronics12071627.
[9] A. Robles-Guerrero, T. Saucedo-Anaya, E. González-Ramérez, y C. E. Galván-Tejada, «Frequency Analysis of Honey Bee Buzz for Automatic Recognition of Health Status: A Preliminary Study», *Res. Comput. Sci.*, vol. 142, n.º 1, pp. 89-98, dic. 2017, doi: 10.13053/rcs-142-1-9.
[10] «Smart Bee Colony Monitor: Clips of Beehive Sounds». Accedido: 7 de agosto de 2023. [En línea]. Disponible en: https://www.kaggle.com/datasets/annajyang/beehive-sounds/data
[11] A. Robles-Guerrero, T. Saucedo-Anaya, E. Gonzalez, y J. I. de la Rosa, «Queenless honeybee acoustic patterns», vol. 1, ago. 2022, doi: 10.17632/t9prmbmdfn.1.
[12] I. Nolasco, A. Terenzi, S. Cecchi, S. Orcioni, H. L. Bear, y E. Benetos, «Audio-based identification of beehive states», 15 de febrero de 2019, *arXiv*: arXiv:1811.06330. doi: 10.48550/arXiv.1811.06330.
[13] L. Barbisan y F. Riente, «Machine Learning Framework for the Acoustic Detection of the Queen Bee Presence», en *Proceedings of the 10th Convention of the European Acoustics Association Forum Acusticum 2023*, Turin, Italy: European Acoustics Association, ene. 2024, pp. 4347-4350. doi: 10.61782/fa.2023.1309.
[14] T. Cejrowski, J. Szymanski, H. Mora, y D. Gil, «Detection of the Bee Queen Presence Using Sound Analysis», 2018, pp. 297-306. doi: 10.1007/978-3-319-75420-8_28.