UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE INGENIERÍA EN SISTEMAS, ELECTRÓNICA E INDUSTRIAL

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y COMUNICACIONES

TEMA:

"SISTEMA DE MONITOREO APÍCOLA MEDIANTE EL USO DE REDES NEURONALES ARTIFICIALES PARA IDENTIFICAR LA VARIACIÓN DE POBLACIÓN."

Trabajo de graduación, modalidad: Proyecto de Investigación, presentado previo a la obtención del título de Ingeniero en Electrónica y Comunicaciones.

LÍNEA DE INVESTIGACIÓN: Comunicaciones Inalámbricas.

AUTOR: David Andrés Gavilanes Proaño

TUTOR: Ing. Santiago Altamirano Meléndez

AMBATO - ECUADOR

SEPTIEMBRE - 2020

APROBACIÓN DEL TUTOR

En calidad de tutor del Trabajo de Titulación con el tema: "SISTEMA DE MONITOREO APÍCOLA MEDIANTE EL USO DE REDES NEURONALES ARTIFICIALES PARA IDENTIFICAR LA VARIACIÓN DE POBLACIÓN", desarrollado bajo la modalidad Proyecto de Investigación por el señor David Andrés Gavilanes Proaño, estudiante de la Carrera de Ingeniería en Electrónica y Comunicaciones de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, de la Universidad Técnica de Ambato, me permito indicar que el estudiante ha sido tutorado durante todo el desarrollo del trabajo hasta su conclusión, de acuerdo a lo dispuesto en el Artículo 15 del Reglamento para obtener el Título de Tercer Nivel, de Grado de la Universidad Técnica de Ambato, y el numeral 7.4 del respectivo instructivo.

Ambato, Septiembre 2020

Firmado electrónicamente por:
SANTIAGO MAURICIO ALTAMIRANO MELENDEZ

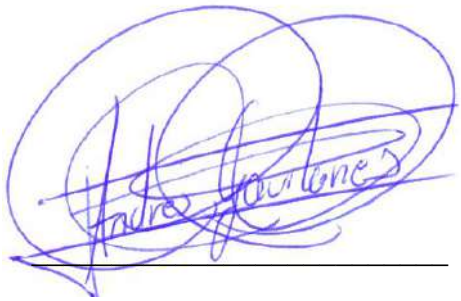
Ing. Santiago Altamirano, Mg.

TUTOR

DERECHOS DE AUTOR

Autorizo a la Universidad Técnica de Ambato, para que haga uso de este Trabajo de Titulación como un documento disponible para la lectura, consulta y procesos de investigación. Cedo los derechos de mi Trabajo de Titulación en favor de la Universidad Técnica de Ambato, con fines de difusión pública. Además, autorizo su reproducción total o parcial dentro de las regulaciones de la institución.

Ambato, Septiembre 2020



David Andrés Gavilanes Proaño.

C.C. 180389885-5

AUTOR

APROBACIÓN TRIBUNAL DE GRADO

En calidad de par calificador del Informe Final del Trabajo de Titulación presentado por el señor David Andrés Gavilanes Proaño, estudiante de la Carrera de Ingeniería en Electrónica y Comunicaciones, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, bajo la Modalidad Proyecto de Investigación, titulado "SISTEMA DE MONITOREO APÍCOLA MEDIANTE EL USO DE REDES NEURONALES ARTIFICIALES PARA IDENTIFICAR LA VARIACIÓN DE POBLACIÓN", nos permitimos informar que el trabajo ha sido revisado y calificado de acuerdo al Artículo 17 del Reglamento para obtener el Título de Tercer Nivel, de Grado de la Universidad Técnica de Ambato, y al numeral 7.6 del respectivo instructivo. Para cuya constancia suscribimos, conjuntamente con la señora Presidenta del Tribunal.

Ambato, Septiembre 2020

Firmado electrónicamente por:
ELSA PILAR URRUTIA

Ing. Pilar Urrutia, Mg.
PRESIDENTA DEL TRIBUNAL

Firmado electrónicamente por:
MARCO ANTONIO JURADO LOZADA

Ing. Marco Jurado, Mg.
PROFESOR CALIFICADOR

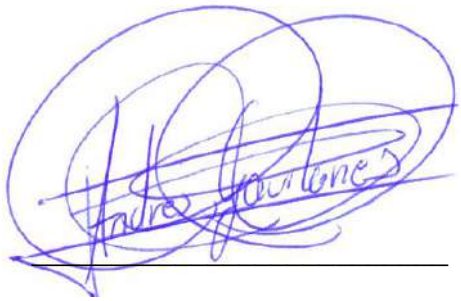
Firmado electrónicamente por:
VICTOR SANTIAGO MANZANO VILLAFUERTE

Ing. Santiago Manzano, Mg.
PROFESOR CALIFICADOR

DERECHOS DE AUTOR

Autorizo a la Universidad Técnica de Ambato, para que haga uso de este Trabajo de Titulación como un documento disponible para la lectura, consulta y procesos de investigación. Cedo los derechos de mi Trabajo de Titulación en favor de la Universidad Técnica de Ambato, con fines de difusión pública. Además, autorizo su reproducción total o parcial dentro de las regulaciones de la institución.

Ambato, Septiembre 2020



David Andrés Gavilanes Proaño.

C.C. 180389885-5

AUTOR

Índice general

Índice de figuras

Índice de tablas

RESUMEN EJECUTIVO

La implementación del sistema de monitoreo apícola mediante el uso de redes neuronales artificiales para identificar la variación de población permite predecir a futuro las variables físicas que afectan directamente la salud y la producción de población de la colmena. En el presente proyecto muestra el desarrollo de un prototipo de sistema de monitoreo apícola, se identifican los rangos óptimos de las variables físicas de temperatura, humedad, concentración de dióxido de carbono y peso de la colmena para asegurar la estabilidad dentro del colmenar, a demás del por qué el uso de los sensores seleccionados para realizar la lectura de dichas variables que fueron elegidos y cómo la información es adquirida y enviada por medio de tecnología inalámbrica WI-FI para ser almacenados en una base de datos MySql, la información es procesada para el desarrollo del modelo de red neuronal artificial de memoria a corto y largo plazo, el cual permite pronosticar en el que dadas las condiciones en las que se encuentra la colmena de los días u horas anteriores, predice las condiciones a los días u horas siguientes, reduciendo la intervención del hombre sobre la colmena, evitando posibles daños en su ecosistema y ofrece información en tiempo real sobre algún cambio significativo en su entorno favoreciendo al usuario o apicultor al tomar medidas preventivas para mejorar el rendimiento de la colmena.

ABSTRACT

The implementation of the beekeeping monitoring system through the use of artificial neural networks to identify population variation allows to predict in the future the physical variables that directly affect the health and population production of the hive. In this project shows the development of a prototype bee monitoring system, the optimal ranges of the physical variables of temperature, humidity, carbon dioxide concentration and weight of the hive are identified to ensure stability within the apiary, as well as why the use of the selected sensors to read the said variables that were chosen and how the information is acquired and sent through WI-FI wireless technology to be stored in a MySql database, the information is processed to the development of the artificial neural network model of short and long-term memory, which allows forecasting in which, given the conditions in which the hive is located in the previous days or hours, it predicts the conditions in the following days or hours, reducing the intervention of man on the hive, avoiding possible damage to its ecosystem and offers information in real time on any changes sig Significant in its environment favoring the user or beekeeper when taking preventive measures to improve the performance of the hive.

Capítulo I

MARCO TEÓRICO

1.1 Antecedentes Investigativos

Al finalizar la investigación basados en artículos científicos, investigación de proyectos realizados a nivel comercial y académica se pueden encontrar los siguientes antecedentes:

Piotr Medrzycki, Fabio Sgolastra, Laura Bortolotti, Gherardo Bogo, Simone Tosi, Erica Padovani, Claudio Porrini y Anna Gloria Sabatini en The International bee research associationn en el año 2009, desarrollo *"Influence of brood rearing temperature on honey bee development and susceptibility to poisoning by pesticides"*, en el cual mostraron los resultados que influye la disminución de la temperatura de la cría de 2°C en relación a los efectos sobre la mortalidad larvaria y adulta, además de la susceptibilidad a la intoxicación por pesticidas, las larvas de la abeja fueron criadas a dos temperaturas distintas: 35°C (óptimo) y 33°C (subóptimo), desde 12h después de la incubación hasta los 15 días de edad, el dimetoato que es un tipo de insecticida fue administrado a las larvas y a la abeja adulta, donde la longevidad fue el parámetro medido, concluyendo que las abejas adultas que se derivaban de la cría mantenida a temperatura subóptima tienen una aptitud más baja y son más susceptibles a la intoxicación por pesticida [12].

Armands Kviesis en el Department of Computer Systems, Faculty of Information Technology, Latvia University of Agriculture, en el año 2016, desarrollo la *"Application of neural networks for honey bee colony state identification"*, En este artículo, los autores proponen un método para la detección del estado de colonias de abejas (inicio del período de cría de cría y enjambres) mediante redes neuronales

con aprendizaje supervisado. Esto se puede lograr mediante la inspección de los datos de temperatura y el desarrollo de algoritmos para cada estado de colonia de abejas o mediante la aplicación de redes neuronales. Las redes neuronales se utilizan ampliamente para diversas tareas, incluidas las relacionadas con la clasificación y el procesamiento de datos [13].

Vyacheslav G. Rybin, en Department of Computer Aided Design, Saint Petersburg Electrotechnical University "LETI" en el año 2017 desarrollo *"Embedded Data Acquisition System for Beehive Monitoring"*, en el cual desarrollaron un sistema de adquisición de datos integrado para el monitoreo automatizado de la colmena. Se proporciona una descripción de los subsistemas de sensores construidos. La solución propuesta adquiere la temperatura, la humedad y el peso de la colmena al referir estos datos a la aplicación móvil a través de una red inalámbrica. El sistema también realiza un análisis de los ruidos de abejas recolectados con redes neuronales artificiales para reconocer las tuberías de abejas e informa al apicultor para evitar el enjambre no deseado. Se representan los esquemas y los algoritmos del sistema propuesto, así como los resultados experimentales obtenidos [14].

Aldis Pecka, Vitalijs Osadcuks, en Faculty of Engineering, Latvia University of Agriculture, Latvia, en el año 2017, desarrollo *"Development of Internet of Things concept for Precision Beekeeping"* , esta investigación presenta el diseño conceptual del enfoque de Internet of Thing para la apicultura. Además, ha desarrollado un sistema de monitoreo de la temperatura de la colonia de abejas basado en la red local de sensores inalámbricos junto con la interfaz externa GSM / GPRS para la comunicación basada en paquetes con el servidor remoto en Internet [15].

1.2 Contextualización del problema

El desorden del colapso de la colonia (o Colony Collapse Disorder, CCD, por sus siglas en inglés) es un fenómeno en el que las abejas obreras de una colmena desaparecen bruscamente de manera instantánea y sin ninguna razón obvia, abandonando así a

su reina, y rompiendo con la dinámica habitual de un enjambre natural, donde un gran grupo de abejas obreras siguen a la reina para conformar una nueva colonia, los posibles factores que afectan al CCD están amenazadas por: agentes patógenos, parásitos, estrés derivado del ambiente, falta de hábitat natural, aumento de la exposición a sustancias químicas artificiales, por el mal manejo de las colmenas, entre otros, que ejercen efectos claros y negativos sobre la salud de los polinizadores, tanto individual y al nivel de colonia, afectando la producción y comercialización de miel, polen propóleo, jalea real y cera [16].

El Ministerio de Agricultura y Ganadería (MAG), determinó que en el Ecuador existen un déficit de producción en el mercado, el Programa Nacional de Apicultura (Pronapis), informó que la producción promedio de miel es de 10,2 kilogramos por colmena al año pero, busca que el país aumente su promedio nacional a 15,2 kilogramos por colmena al año. En el Ecuador existen 1.760 apicultores y 19.155 colmenas registradas hasta Julio de 2018, que proveen al mercado nacional de miel de abeja, polen, propóleo y cera, que en comparación con los datos registrados en la década de los noventa del siglo pasado, hasta 1993, se registraron 38.500 colmenas, años posteriores la actividad decayó debido a la desaparición de los programas apícolas gubernamentales [17].

En un estudio del Fondo Internacional de Desarrollo Agrícola (FIDA), elaborado en 2015, se indica que Ecuador consume 601 toneladas de miel por año, pero apenas produce 200. Debido a esto, según datos del Banco Central del Ecuador (BCE), entre 2000 y 2011 se importaron 1.615 toneladas métricas de miel, con precios que oscilaron entre $4.500 y $5.030 por tonelada. Andrés Miño, coordinador del Pronapis, afirmó que es necesario crear proyectos que impulsen al crecimiento de producción de miel de abeja en el país, por tal motivo el MAG busca crear plataformas donde se registre información real, veraz y actualizada relacionada con la producción de miel en el país [18], [19].

La necesidad de crear un sistema de monitoreo es fundamental en el desarrollo de una colmena debido a que el comportamiento del sistema biológico de las abejas y su reacción a la intervención humana es impredecible, al aplicar técnicas de monitorización permitirá evaluar la incidencia sobre las colonias en condiciones ambientales climáticas y su relación con el estado de las abejas, de esta manera ayudar a los apicultores a evitar altas mortalidades de abejas para mejorar el rendimiento de producción de polen, jalea real, cera y miel [20].

El sistema de monitoreo apícola generará beneficios a los apicultores quienes será capaz de recopilar datos discretamente de una colmena, que describe las condiciones y las actividades de una colonia de abejas, implementando una amplia gama de sensores para monitorear las condiciones multidimensionales dentro de una colmena viva controlando temperatura, humedad, concentración de dióxido de carbono, sonido y peso del interior de las colmenas, entregando alertas tempranas a los apicultores para su respectiva revisión, con base a los datos recopilados, el sistema podrá estimar la producción de miel, la salud de las abejas y los posibles factores que afectan al desorden del colapso de la colonia.

1.3 Fundamentación Teórica

1.3.1 Apicultura

La apicultura es la actividad dedicada a la crianza de las abejas en cavidades en las que pueden construir panales, poner huevos y, lo más importante, producir miel. La estructura que alberga esta cavidad se conoce como la colmena. Cuando las abejas melíferas reciben en una colmena ideal, pueden producir más miel de la que necesitan para sobrevivir, dejando un excedente para los apicultores. Por lo tanto, la apicultura, por su propia naturaleza, está fundamental e inextricablemente vinculada al diseño de la colmena [1].

1.3.2 Castas de la abeja melífera

Las abejas melíferas son insectos sociales con un sistema de castas de individuos. En cada colonia hay una hembra reproductora llamada reina como se muestra en la Figura 1.1a, cuyas estructuras reproductivas maduran como resultado de una dieta especial alimentada durante las etapas larvarias. En la cima de su fertilidad, una abeja reina puede poner hasta 2,000 huevos por día. Los huevos fertilizados se convierten en abejas obreras, y las que no son fertilizadas en machos llamados zánganos [1].

Las abejas obreras se muestra en la Figura 1.1b, cuidan de la progenie de la reina, limpian la colmena, buscan comida en forma de polen, néctar, y producen miel. El papel que desempeñan las abejas obreras en la colonia está determinado en parte por su edad. Cuando la abeja obrera emerge recientemente, permanece dentro de la colmena, limpiando celdas, alimentando larvas y produciendo panales y miel. A medida que madure, se convertirá en una abeja en busca de néctar y polen. La única función de la abeja zángano macho que se muestra en la Figura 1.1c es aparearse con una reina. En general, una colmena saludable tiene mucho menos zánganos que obreras, y representan solo el 5 % de las abejas en la colmena [1].

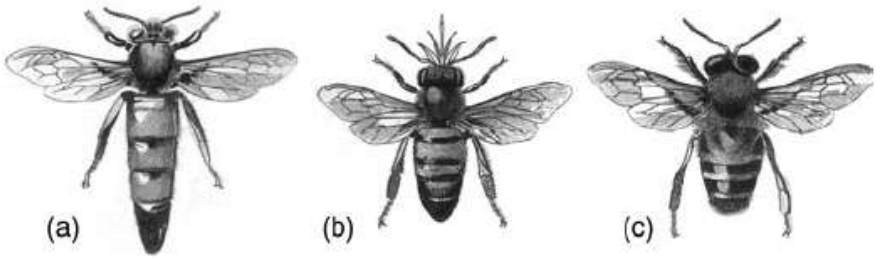


Figura 1.1: Las castas de las abejas melíferas: (a) reina, (b) obrera, (c) zángano [1].

1.3.3 Apiario o colmenar

Un apiario es el lugar donde la abeja melífera construye su nido en una cavidad, generalmente en un árbol hueco, en una grieta entre rocas o en una madriguera abandonada en el suelo. En él, las abejas construyen panales hechos de cera secretada por las abejas, que cuelgan verticalmente desde la parte superior, con dos capas de celdas horizontales, una detrás de otra, en ángulo de aproximadamente 13° para evitar

que la miel se derrame en las aberturas opuestas. Los panales están construidos con células hexagonales, en las cuales la miel se almacena en la parte superior y el polen entre la miel y la cría [21].

1.3.4 Ciclo de vida

El ciclo de vida de las abejas melífera pasan por cuatro etapas: huevo, larva, pupa y adulto. De la reina pone los huevos, que son de color blanco perla, cilíndricos, ovalados y alargados de entre 1.3 y 1.8 mm de largo, pueden desarrollarse como obreras o zánganos, dependiendo de si los huevos están fertilizados o no. Después de 3 días, una larva emerge del huevo, que será alimentada progresivamente por las abejas nodrizas, durante los primeros 3 días con jalea real, y después con una mezcla de miel y polen. Solo las larvas de la cría reina reciben jalea real durante la etapa larval completa. Al final de esta etapa, durante la cual las células están sin tapar, las larvas giran un capullo y se transforman en pupas después de que los trabajadores adultos han tapado las células. Durante la etapa pupal, la cutícula se oscurece gradualmente, y los músculos y órganos internos cambian a sus formas adultas, antes de que tenga lugar la muda final a la etapa adulta. Finalmente, el adulto comienza a quitar el tapón de la celda de adentro hacia afuera usando sus mandíbulas y emerge de la celda, despliega sus alas y antenas, y comienza sus actividades. La metamorfosis del huevo al adulto es más corta para las reinas, demora tan solo 16 días en apis mellifera y más larga para los zánganos con una duración de 24 días. Para las obreras, se necesitan aproximadamente 21 días para desarrollarse del huevo a la etapa adulto, en la Figura 1.2 se muestra el desarrollo diario del ciclo de vida de las castas de abejas melíferas [21].

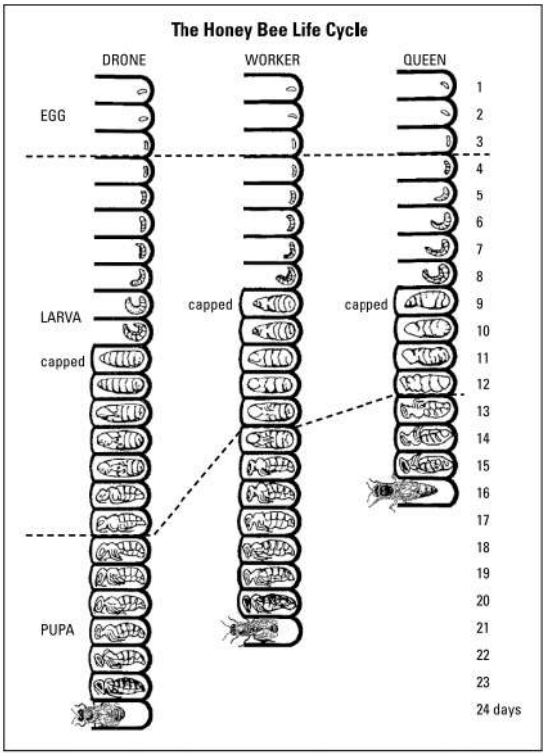


Figura 1.2: Ciclo de desarrollo diario de las dos castas femeninas y el zángano, desde el huevo hasta el adulto [2].

1.3.5 Inteligencia Artificial

La inteligencia artificial (IA) como se muestra en la Figura 1.3 es un campo general que abarca el aprendizaje automático y el aprendizaje profundo, pero que también incluye muchos más enfoques que no implican ningún aprendizaje. Durante un tiempo bastante largo, varios expertos creyeron que la inteligencia artificial a nivel humano podría lograrse, que los programadores elaboren un conjunto suficientemente grande de reglas explícitas para manipular el conocimiento. Este enfoque se conoce como IA simbólica, y fue el paradigma dominante en IA desde la década de 1950 hasta finales de la década de 1980. Alcanzó su máxima popularidad durante el auge de los sistemas expertos de la década de 1980. Aunque la IA simbólica demostró ser adecuada para resolver problemas lógicos bien definidos, como jugar al ajedrez, resultó ser difícil de resolver reglas explícitas para resolver problemas más complejos y difusos, como la clasificación de imágenes, el reconocimiento de voz y la traducción de idiomas. Surgió un nuevo enfoque para tomar el lugar simbólico de la IA: el aprendizaje automático [3].

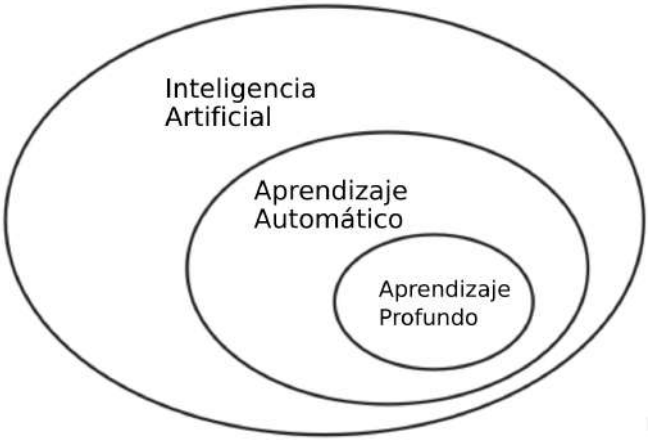


Figura 1.3: Inteligencia artificial, aprendizaje automático, y aprendizaje profundo [3].

1.3.6 Aprendizaje automático

El aprendizaje automático (o Machine Learning, ML, por sus siglas en ingles) es un subcampo de la informática que evolucionó a partir del estudio del reconocimiento de patrones y la teoría del aprendizaje computacional en Inteligencia Artificial (IA). Una computadora con capacidades de aprendizaje automático puede mejorar su rendimiento en función de su propia experiencia sin tener un programa explícito que les diga exactamente qué hacer. Las técnicas de aprendizaje automático se pueden dividir en las siguientes clases: aprendizaje supervisado, aprendizaje sin supervisión y aprendizaje reforzado [22].

El aprendizaje automático da la posibilidad de resolver problemas más complejos con una mínima interferencia humana. Dependiendo del tipo de aprendizaje automático que utilice, una vez que esté activo en el sistema, la máquina puede continuar haciendo sus propios ajustes y reconocer sus propios fallos y éxitos [22].

Canalización del aprendizaje automático

La canalización del aprendizaje automático es la estructura de la creación de la inteligencia artificial que permite resolver problemas del mundo real mediante un proceso estructurado. Al resolver un problema de análisis o aprendizaje automático del mundo real se comienza desde la obtención de los datos hasta transformarlos en información mediante algoritmos y técnicas de aprendizaje automático, que

principalmente están relacionados con la recuperación y extracción de datos, la preparación, el modelado, la evaluación y la implementación. En la Figura 1.4 se muestra una visión general y sus fases principales en la canalización de aprendizaje automático que se describen a continuación:

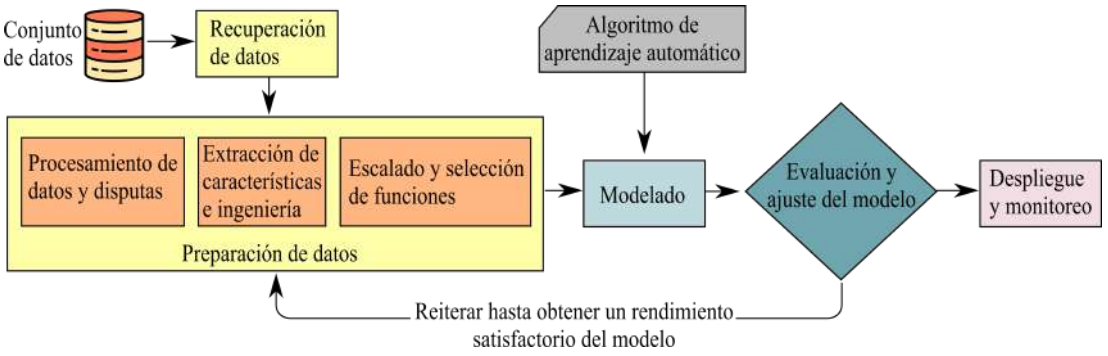


Figura 1.4: Tubería estándar de aprendizaje automático [4].

• **Recuperación de datos:** La recuperación de datos es la recopilación, extracción y adquisición de varias de fuentes y almacenes de datos.

• **Preparación de datos:** La preparación de datos prepara, limpia, refuerza y manipula los datos según sea necesario.

• **Procesamiento de datos y discusión:** El procesamiento de datos y discusión se ocupa del procesamiento de datos, limpieza, organización y realización de análisis de datos descriptivos y exploratorios iniciales.

• **Extracción de características e ingeniería:** La extracción de características e ingeniería se encarga de extraer los atributos mas importantes de los datos sin procesar e incluso se crea o diseña nuevas características a partir de entidades existentes.

• **Escalado y selección de funciones:** El escalado y selección de funciones de los datos a menudo deben normalizarse y escalarse para evitar que los algoritmos de aprendizaje automático se sesguen. Además se necesita seleccionar un subconjunto de todas las funciones disponibles en función de la importancia y calidad de las funciones.

• **Modelado:** El modelado provee las características de los datos a un método o algoritmo de Machine Learning y capacitar al modelo, generalmente para optimizar una función de costo específica en la mayoría de los casos con el objetivo de reducir errores y generalizar las representaciones aprendidas de los datos.

• **Evaluación y ajuste del modelo:** La evaluación y ajuste del modelo evalúan y aprueban el conjuntos de datos de validación, en función de métricas como precisión, puntuación y otros, que evalúan el rendimiento del modelo. Los modelos tienen varios parámetros que se ajustan en un proceso llamado optimización de hiperparámetros para obtener modelos con los mejores y óptimos resultados.

• **Despliegue y monitoreo:** El despliegue y monitore de los modelos seleccionados se implementan en producción y se supervisan constantemente según sus predicciones y resultados [4].

1.3.7 Aprendizaje supervisado

El aprendizaje supervisado es un tipo de aprendizaje en el que los datos de entrenamiento se etiquetan con descripciones o valores, los métodos o algoritmos de aprendizaje supervisados incluyen algoritmos de aprendizaje que toman muestras de datos (conocidos como datos de entrenamiento) y resultados asociados (conocidos como etiquetas o respuestas) con cada muestra de datos durante el proceso de entrenamiento del modelo. El objetivo principal es aprender un mapeo o asociación entre las muestras de datos de entrada *'x'* y sus salidas correspondientes *'y'* basadas en múltiples instancias de datos de entrenamiento. Este conocimiento aprendido se puede utilizar en el futuro para predecir una salida y' para cualquier nueva muestra de datos de entrada x' que antes se desconocía o no se veía durante el proceso de capacitación del modelo. Estos métodos se denominan supervisados porque el modelo aprende sobre muestras de datos donde las respuestas/etiquetas de salida deseadas ya se conocen de antemano en la fase de capacitación [4].

En el aprendizaje supervisado, la máquina ya está programada para esperar una cierta salida de un algoritmo en su sistema antes de comenzar su trabajo. En esencia, sabe el tipo de respuesta que está tratando de alcanzar, y simplemente necesita resolver los diferentes pasos necesarios para encontrarla. El algoritmo lo aprende un conjunto especializado de datos de entrenamiento que "guía" la máquina a la conclusión correcta. Entonces, si algo sale mal y los algoritmos producen un resultado que es muy diferente del resultado esperado, los datos de entrenamiento ingresados

previamente intervendrán y redirigirán las funciones para que la computadora vuelva a la normalidad [22].

Canalización supervisada del aprendizaje automático

La canalización supervisada de aprendizaje automático se tratan de trabajar con datos etiquetados supervisados para entrenar modelos y luego predecir los resultados de nuevas muestras de datos. Algunos procesos como la ingeniería de características, el escalado y la selección siempre deben permanecer constantes para que se usen las mismas características para entrenar el modelo y se extraigan las mismas características de nuevas muestras de datos para alimentar el modelo en la fase de predicción. Basado en la canalización genérica de aprendizaje automático anterior, la Figura 1.5 se muestra una canalización supervisada de *Machine Learning* estándar [4].

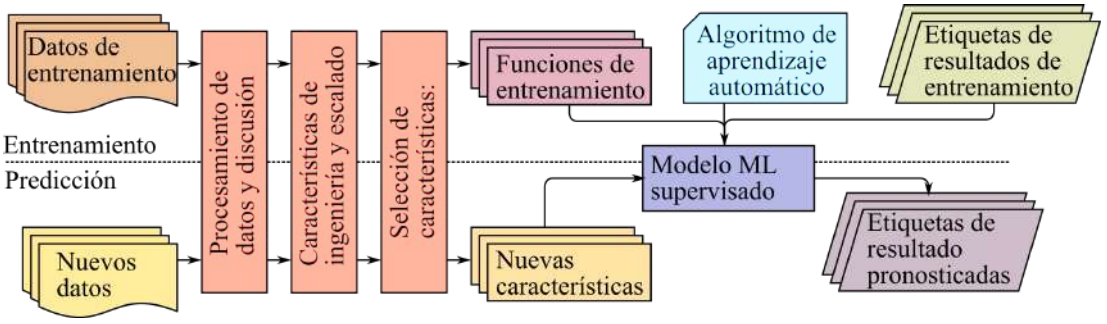


Figura 1.5: Canalización supervisada de aprendizaje automático [4].

La misma secuencia de procesamiento de datos, disputas, ingeniería de características, escalado y selección se utiliza tanto para los datos utilizados en el entrenamiento del modelo como para las muestras de datos futuros para los cuales el modelo predice resultados. Este es un punto muy importante que debe recordar siempre que construya un modelo supervisado. Además de esto, como se muestra, el modelo es una combinación de un algoritmo de aprendizaje automático (supervisado) y características de datos de entrenamiento y etiquetas correspondientes. Este modelo tomará características de nuevas muestras de datos y generará etiquetas pronosticadas en la fase de predicción. Los métodos de aprendizaje supervisados son de dos clases principales basadas en el tipo de tareas de aprendizaje automático que pretenden resolver: clasificación y regresión [4].

Clasificación La clasificación es un subtipo de aprendizaje supervisado donde las etiquetas representan clases, con el objetivo de predecir etiquetas de salida o respuestas de naturaleza categórica para los datos de entrada en función de lo que el modelo ha aprendido en la fase de capacitación. Las etiquetas de salida se les conocen como clases o etiquetas de clase de naturaleza categórica, lo que significa que son valores no ordenados y discretos [4].

Regresión La regresión es un subtipo de aprendizaje supervisado cuando los datos de entrada se etiquetan con números reales, los métodos basados en regresión están entrenados en muestras de datos de entrada que tienen respuestas de salida que son valores numéricos continuos a diferencia de la clasificación, donde se tiene categorías o clases discretas. Los modelos de regresión hacen uso de atributos o características de datos de entrada y sus correspondientes valores numéricos continuos de salida para aprender relaciones y asociaciones específicas entre las entradas y sus salidas correspondientes. El objetivo principal es minimizar los errores durante el entrenamiento y validar el modelo para que se generalice bien, no se sobreajuste o se sesgue solo a los datos del entrenamiento y funcione bien en predicciones futuras [4].

1.3.8 Aprendizaje no supervisado

El aprendizaje no supervisado es descubrir patrones significativos en conjuntos de datos, estos métodos se denominan no supervisados porque el modelo o algoritmo intenta aprender estructuras, patrones y relaciones latentes inherentes a partir de datos dados sin ninguna ayuda o supervisión, como proporcionar anotaciones en forma de resultados o resultados etiquetados [4].

Con el aprendizaje no supervisado, el sistema no cuenta con datos preexistentes y los resultados de los problemas aún no se conocen. En otras palabras, el programa está trabajando a ciegas usando solo operaciones lógicas para trazar su camino hacia una decisión. Esto hace que la resolución de problemas en el aprendizaje no supervisado

sea muy desafiante, pero es este tipo de aprendizaje el que está más estrechamente relacionado con la forma en que la mente humana procesa la información [22].

El aprendizaje no supervisado se preocupa más por tratar de extraer información o información significativa de los datos en lugar de tratar de predecir algunos resultados en función de los datos de capacitación supervisada previamente disponibles [4].

Canalización del aprendizaje automático no supervisada

La canalización del aprendizaje automático no supervisada se enfoca en extraer patrones, relaciones, asociaciones y grupos de datos. Los procesos relacionados con la ingeniería de características, el escalado y la selección son similares al aprendizaje supervisado. Sin embargo, no hay un concepto de datos pre-etiquetados aquí. Por lo tanto, la canalización del aprendizaje automático no supervisada sería ligeramente diferente en contraste con la canalización supervisada. La Figura 1.6 se muestra una canalización estándar de aprendizaje automático sin supervisión [4].

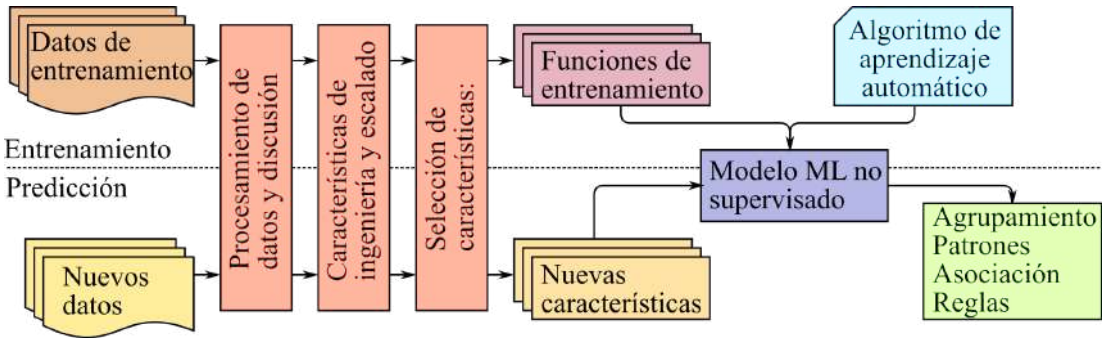


Figura 1.6: Canalización de aprendizaje automático sin supervisión [4].

Los métodos de aprendizaje no supervisados se pueden clasificar en las siguientes áreas amplias de tareas de Machine Learning relevantes para el aprendizaje no supervisado: agrupación, reducción de dimensionalidad, detección de anomalías y asociación de minería de reglas [4].

Agrupación

La agrupación en clústeres es un proceso de búsqueda de grupos de objetos similares en conjuntos de datos que intentan encontrar patrones de similitud y relaciones entre las muestras de datos y luego agrupan estas muestras en varios grupos, de modo que cada grupo o agrupación de muestras de datos tiene alguna similitud, según los atributos o características inherentes en los datos [4].

Reducción de dimensionalidad

En el aprendizaje automático la reducción de dimensionalidad es el proceso de reducción del número de variables aleatorias al extraer atributos o características de muestras de datos sin procesar para cada muestra de datos. Estos métodos reducen el número de variables de características al extraer o seleccionar un conjunto de características principales o representativas [4].

Detección de anomalías

La detección de anomalías es un grupo de técnicas utilizadas para identificar comportamientos inusuales que no cumplen con el patrón de datos esperado como se muestra en la Figura 1.7, las anomalías ocurren con poca frecuencia y, por lo tanto, son eventos raros, y en otros casos, las anomalías pueden no ser raras, pero pueden ocurrir en ráfagas muy cortas con el tiempo, por lo que tienen patrones específicos. Los métodos de aprendizaje no supervisados se pueden utilizar para la detección de anomalías, de modo que sen entrena el algoritmo en el conjunto de datos de entrenamiento con muestras de datos normales y no anómalas. Una vez que aprende los datos necesarios representaciones, patrones y relaciones entre atributos en muestras normales, para cualquier nueva muestra de datos, sería capaz de identificarlo como un punto de datos anómalo o normal utilizando su conocimiento aprendido [4].

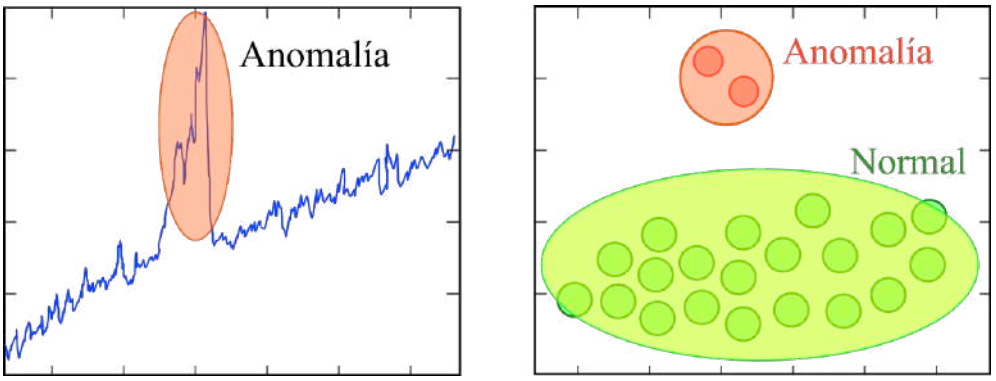


Figura 1.7: Aprendizaje no supervisado: detección de anomalías [4].

Asociación de Minería de Reglas

La minería de reglas de asociación es un método de minería de datos que se usa para examinar y analizar grandes conjuntos de datos transaccionales para encontrar patrones y reglas de interés. Estos patrones representan relaciones y asociaciones interesantes, entre varios artículos entre transacciones. La minería de reglas de asociación también se denomina análisis de la cesta de la compra, que se utiliza para analizar los patrones de compra de los clientes. Las reglas de asociación ayudan a detectar y predecir patrones transaccionales basados en el conocimiento que obtiene de las transacciones de capacitación [4].

1.3.9 Aprendizaje reforzado

El aprendizaje reforzado es otro tipo de algoritmos de Machine Learning en el que el objetivo es el desarrollo de un sistema que recibe el nombre de agente que se desea que mejore su eficiencia realizando cierta tarea basándose en la interacción con su entorno para interactuar y mejorar su rendimiento durante un período de tiempo con respecto al tipo de acciones que realiza en el entorno. Normalmente, el agente comienza con un conjunto de estrategias o políticas para interactuar con el entorno. Al observar el medio ambiente, toma una acción particular basada en una regla o política y observando el estado actual del medio ambiente. Según la acción, el agente obtiene una recompensa, que podría ser beneficiosa o perjudicial en forma de penalización. Actualiza sus políticas y estrategias actuales si es necesario y este proceso iterativo continúa hasta que aprende lo suficiente sobre su entorno para obtener las recompensas

deseadas [4].

Los algoritmos ayudan a las máquinas a aprender en función del resultado de las decisiones que toman. Es un sistema complejo que se basa en una gran cantidad de algoritmos diferentes que trabajan juntos para determinar qué sucede a continuación para lograr los resultados deseados o para resolver un problema específico [22].

1.3.10 Aprendizaje basado en modelos

Los métodos de aprendizaje basados en modelos son un enfoque del aprendizaje automático más tradicional hacia la generalización basada en datos de capacitación. Normalmente, se lleva a cabo un proceso iterativo en el que los datos de entrada se utilizan para extraer características y los modelos se crean en función de varios parámetros del modelo (conocidos como hiperparámetros). Estos hiperparámetros están optimizados en función de varias técnicas de validación de modelos para seleccionar el modelo que mejor se generaliza en los datos de entrenamiento y cierta cantidad de datos de validación y prueba. Finalmente, el mejor modelo se usa para hacer predicciones o decisiones cuando sea necesario [4].

1.3.11 Aprendizaje profundo

También conocido como Deep Learning, que es un subcampo del aprendizaje automático relacionado con algoritmos inspirados en la estructura y función del cerebro llamadas redes neuronales artificiales. Debido a su versatilidad y adaptabilidad en tantas situaciones, ha permitido que las computadoras puedan detectar patrones de voz, crear programas de texto a voz, recuperar información cuando sea necesario e incluso predecir el uso del consumidor en diferentes industrias, el término profundo se refiere a la profundidad de la arquitectura de la red neuronal artificial, y el aprendizaje significa aprender a través de la propia red neuronal artificial [22], [6].

1.3.12 Red neuronal artificial

Una red neuronal artificial (ANN) es una red computacional inspirada en redes neuronales biológicas como un sistema de nodos interconectados entre si, que son redes complejas de neuronas en cerebros humanos. Los nodos creados en el ANN supuestamente están programados para comportarse como neuronas reales y, por lo tanto, son neuronas artificiales. En la Figura 1.8 se muestra la red de nodos (neuronas artificiales) que forman la red neuronal artificial. [5]

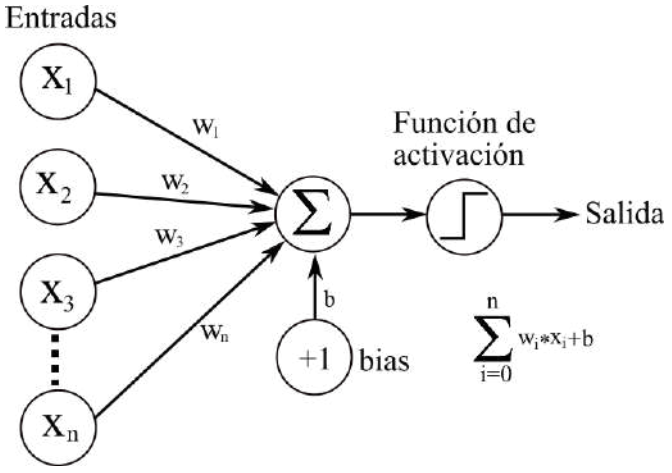


Figura 1.8: Red Neuronal Artificial [5]

El número de capas y el número de neuronas/nodos por capa pueden ser el componente estructural principal de una red neuronal artificial. Una red neuronal típica esta comprendida de las siguientes partes: capas de entradas, capas ocultas, neuronas, capas de salida y el algoritmo de entrenamiento [5].

Las redes neuronales están conectadas, lo que significa que cada neurona en la capa oculta está completamente conectada a cada neurona en la capa de entrada anterior y a su siguiente capa de salida. Una red neuronal aprende ajustando los pesos y sesgos en cada capa de forma iterativa para obtener los resultados óptimos [5].

Capa de entrada

La capa de entrada es el punto de ingreso de una red neuronal para las entradas que está dando al modelo. No hay neuronas en esta capa porque su propósito principal es servir como un conducto hacia las capas ocultas. No se realiza ningún cálculo en ninguno de los nodos de entrada, simplemente transmiten la información a los nodos ocultos. [23].

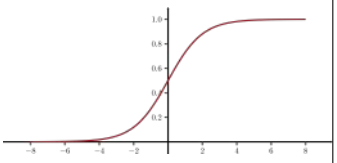
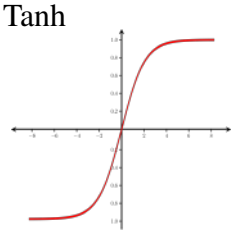
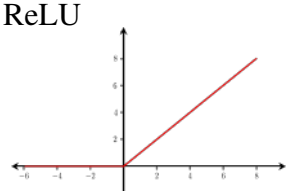
Capas ocultas

Las capas ocultas permiten que las redes neuronales modelen datos no lineales, cada capa oculta contiene un conjunto de neuronas, que luego pasan a la capa de salida.Sin capas ocultas, las redes neuronales serían un conjunto de combinaciones lineales ponderadas, los nodos ocultos realizan cálculos y transfieren información de los nodos de entrada a los nodos de salida. [23].

Neuronas

Las neuronas son combinaciones lineales ponderadas que se envuelven en una función de activación. La combinación lineal ponderada (o suma) es una forma de agregar todos los datos de las neuronas anteriores en una salida para que la siguiente capa se consuma como entrada. Las funciones de activación, que se muestran en la Tabla 1.1, sirven como una forma de normalizar los datos. El propósito de la función de activación es introducir la no linealidad en la salida de una neurona. Esto es importante porque la mayoría de los datos del mundo real no son lineales y se requiere que las neuronas aprendan estas representaciones no lineales. A medida que una red este enviando información hacia adelante, está agregando entradas anteriores en sumas ponderadas, recibe y calcula el valor activado en función de una función de activación [23].

Tabla 1.1: Funciones de activación – redes neuronales.

	Ventajas	Desventajas
Sigmoid/ Logístico  $f(x) = \frac{1}{1+e^{-x}}$	• Gradiente suave, evita "saltos" en los valores de salida. • Los valores de salida se unen entre 0 y 1, normalizando la salida de cada neurona. • Predicciones claras: para X por encima de 2 o por debajo de -2, tiende a llevar el valor Y (la predicción) al borde de la curva, muy cerca de 1 o 0. Esto permite predicciones claras.	• Gradiente de fuga: para valores de X muy altos o muy bajos, casi no hay cambios en la predicción, lo que causa un problema de gradiente de fuga. Esto puede provocar que la red se niegue a aprender más o que sea demasiado lenta para alcanzar una predicción precisa. • Salidas no centradas en cero.
Tanh  $f(x) = \frac{e^{x}-e^{-x}}{e^{-x}+e^{-x}}$	• Centrado en cero : facilita el modelado de entradas que tienen valores muy negativos, neutrales y muy positivos. • De lo contrario, como la función Sigmoide.	• Como la función sigmoidea
ReLU  $f(x) = max(0,x)$	• Computacionalmente eficiente: permite que la red converja muy rápidamente. • No lineal: aunque parece una función lineal, ReLU tiene una función derivada y permite la propagación hacia atrás	• El problema de Dying ReLU: cuando las entradas se acercan a cero o son negativas, el gradiente de la función se convierte en cero, la red no puede realizar la propagación hacia atrás y no puede aprender.

Elaborado por: El investigador, en base a [24].

Las funciones de activación se aplican en múltiples capas de una red. Se utilizan para lograr una alta no linealidad en la entrada, lo que puede ser útil para modelar relaciones complejas entre la entrada y la salida [25].

Capa de salida

La capa de salida es similar a la capa de entrada, excepto que tiene neuronas. Aquí es donde los datos salen del modelo. Las capas de salida deciden cuántas neuronas salen, que es una función de lo que se esta modelando, además son responsables de los cálculos y la transferencia de información de la red al mundo exterior. [23].

Algoritmos de entrenamiento

El algoritmo de entrenamiento es un proceso que modifica el valor de los pesos y bias asociados a cada neurona con el fin de que la red neuronal artificial pueda a partir de unos datos presentados en la entrada, generar una salida. Existen muchos de estos algoritmos, pero los más comunes son: *Back propagation*, *QuickProp* y *RProp*. Un algoritmo de entrenamiento recorre toda la red neuronal y lo compara con lo que se espera. En este punto, aprende de errores de cálculo pasados [23].

1.3.13 Clasificación de redes neuronales

La clasificación de redes neuronales se clasifica según la topología de red en función de sus características más notables:

Redes neuronales de avance

Las redes neuronales de avance constituyen las unidades básicas de la familia de redes neuronales, el movimiento de datos en cualquier red neuronal de avance es desde la capa de entrada a la capa de salida, a través de las capas ocultas actuales, lo que restringe cualquier tipo de bucles como se muestra en la Figura 1.9, la salida de una capa sirve como entrada para la siguiente capa, con restricciones en cualquier tipo de bucles en la arquitectura de red [6].

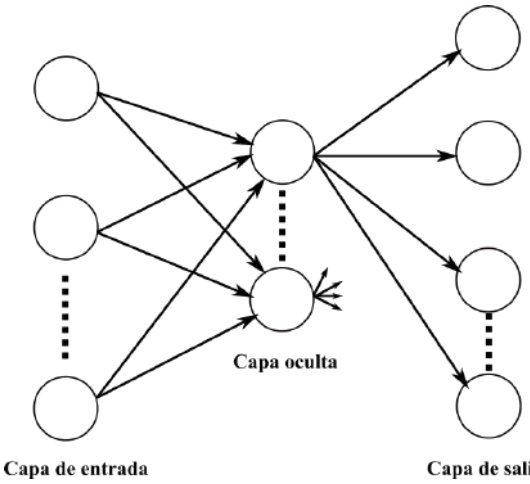


Figura 1.9: Una red neuronal de alimentación de múltiples capas [6].

Redes neuronales convolucionales

Las redes neuronales convolucionales es un algoritmo de Deep Learning que está diseñado para el reconocimiento de imágenes y el reconocimiento de escritura a mano. Su estructura se basa en muestrear una ventana o porción de una imagen, detectar sus características y luego usar las características para construir una representación. Como es evidente en esta descripción, esto lleva al uso de varias capas, por lo tanto, estos modelos fueron los primeros modelos de aprendizaje profundo [6].

Redes neuronales recurrentes

Las redes neuronales recurrentes (RNN) es una red neuronal que integra bucles de realimentación como se muestra en la Figura 1.10, permitiendo a través de ellos que la información persista cuando un patrón de datos cambia con el tiempo. Se puede suponer que las RNNs se desenrollan con el tiempo y se aplican a la misma capa de entrada en cada paso de tiempo utilizando la salida, es decir, el estado de los pasos de tiempo anteriores como entradas [6].

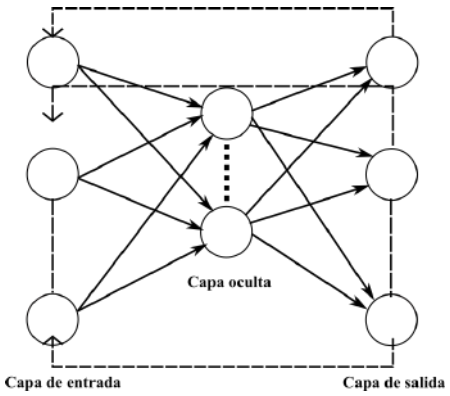


Figura 1.10: Red neuronal recurrente [6].

Las RNNs tienen bucles de retroalimentación en los que la salida del disparo anterior o el índice de tiempo T se alimenta como una de las entradas en el índice de tiempo T + 1. Puede haber casos en los que la salida de la neurona se alimenta a sí misma como entrada. Como estos son muy adecuados para aplicaciones que involucran secuencias, se usan ampliamente en problemas relacionados con videos, que son una secuencia de tiempo de imágenes, y para propósitos de traducción, en donde la comprensión de la siguiente palabra se basa en el contexto del texto anterior. Los siguientes son varios tipos de RNN: [6].

- **Codificación de redes neuronales recurrentes:** La codificación de redes neuronales recurrentes es un conjunto de RNN que toma una entrada en forma de secuencia y genera un estilo de codificación como se muestra en la Figura 1.11, este tipo de red es una parte esencial del sistema de codificación-decodificación utilizado para la traducción de idiomas.

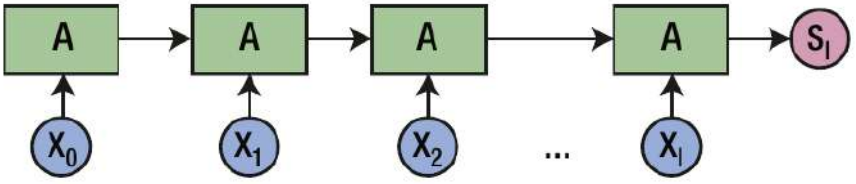


Figura 1.11: Codificación de RNNs [6].

- **Generalización de redes neuronales recurrentes:**: La generalización de redes neuronales recurrentes como se muestra en la Figura 1.12, generan una secuencia de salida de números o valores como las palabras en una oración.

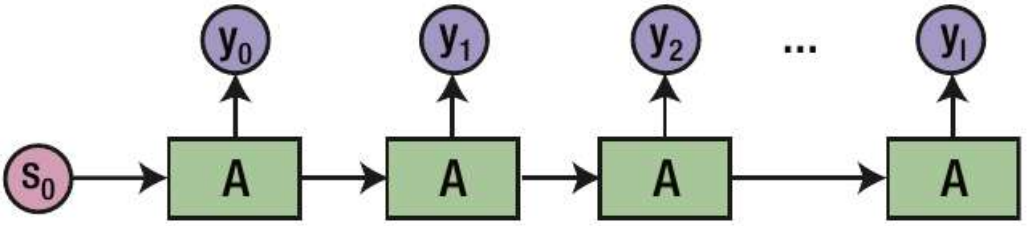


Figura 1.12: Generando RNNs [6]

- **Redes neuronales recurrentes generales**: Las redes neuronales recurrentes generales son una combinación de los dos tipos anteriores de RNN en la Figura 1.13 se muestra la RNN general y se usan para generar secuencias en tareas de NLG (generación de lenguaje natural).

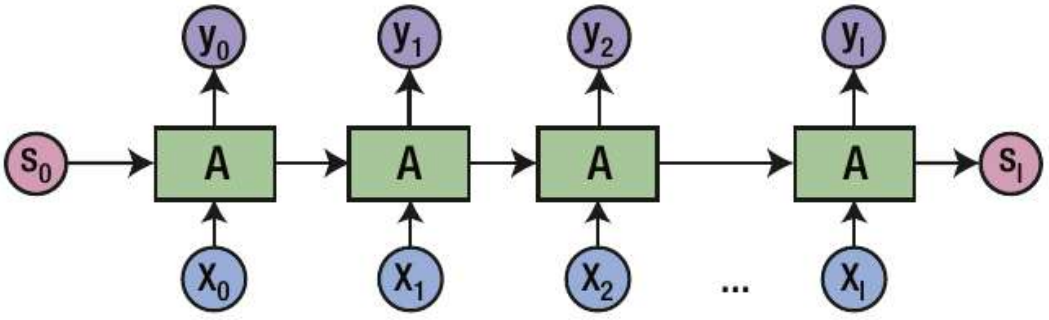


Figura 1.13: General RNNs [6]

Las RNN han demostrado ser la única opción para tratar problemas relacionados con la clasificación de secuencia, y han demostrado ser apropiados para retener la información de los datos de entrada anteriores y utilizar esa información para modificar la salida en cualquier momento. Sin embargo, si la longitud de la secuencia es lo suficientemente larga, entonces los gradientes calculados durante el proceso de entrenamiento del modelo RNN, específicamente la propagación hacia atrás, desaparecen, debido al efecto de multiplicación acumulativa de valores entre 0 y 1, o explotan, nuevamente debido a la multiplicación acumulativa de valores grandes, lo que hace que el modelo se entrene de una manera relativamente lenta [6].

Redes de memoria a corto y largo plazo

Las redes de memoria a corto y largo plazo es una red neuronal recurrente que se entrena utilizando la retropropagación a través del tiempo, y fueron introducidas por primera vez por Sepp Hochreiter y Jürgen Schmidhuber en 1997 y resolvieron el problema de retener información para RNN durante períodos de tiempo más largos. Una red LSTM es el tipo de arquitectura RNN, eso ayuda en el entrenamiento del modelo sobre secuencias largas y en la retención de la memoria de los pasos de tiempo anteriores de entrada alimentados al modelo. Idealmente, resuelve el problema de desaparición de gradiente o explosión de gradiente, al introducir compuertas adicionales, compuertas de entrada y de olvido, que permiten un mejor control sobre el gradiente, permitiendo que información conservar y que olvidar, controlando así el acceso a la información al estado actual de la célula, lo que permite una mejor preservación de las dependencias de largo alcance [6].

Las redes LSTM también tienen una estructura en cadena, pero el módulo de repetición tiene una estructura diferente. En lugar de tener una sola capa de red neuronal, hay cuatro que interactúan de una manera muy especial. La estructura de una celda LSTM se muestra en la Figura 1.14

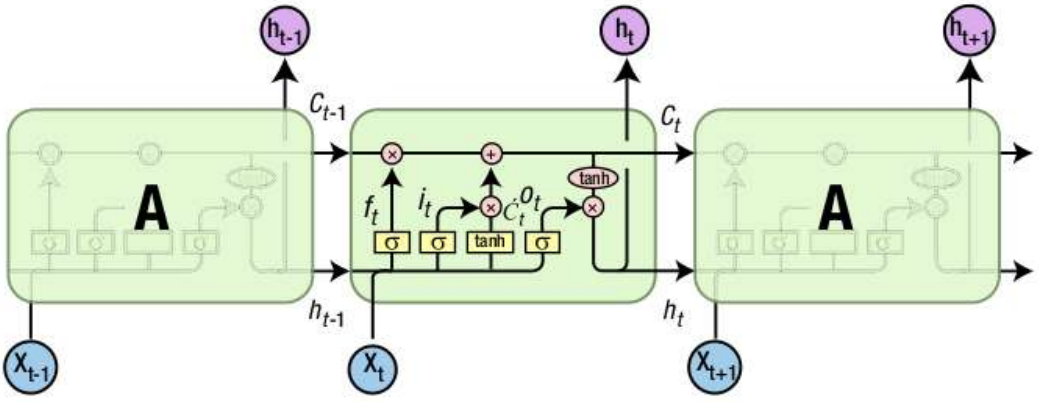


Figura 1.14: Módulo LSTM con cuatro capas interactivas [6].

LSTM se forma utilizando múltiples puertas, que sirven como una buena opción para regular la información que pasa. Tienen una capa de red neuronal sigmoidea, con salida en [0,1] para pesar el límite de paso del componente, y una operación de multiplicación puntual [6].

Redes de codificador-decodificador

Las redes de codificador-decodificador son redes neuronales recurrentes que se ha convertido en un enfoque efectivo y estándar para la predicción de la traducción automática neural (NMT) para la predicción de secuencia a secuencia, se usan para crear una representación interna de la entrada, o para "codificarla", y esa representación se usa como entrada para que otra red produzca la salida. Esto es útil para ir más allá de una clasificación de la entrada. El resultado final puede estar en la misma modalidad, es decir, traducción de idiomas, o en una modalidad diferente, por ejemplo, etiquetado de texto de una imagen, basado en conceptos [6].

Redes neuronales recursivas

Una red neuronal recursiva es un conjunto fijo de pesos que se aplica recursivamente sobre la estructura de la red y se utiliza principalmente para descubrir la jerarquía o estructura de los datos. Mientras que un RNN es una cadena, una red neuronal recursiva toma la forma de una estructura en forma de árbol como se muestra en la Figura 1.15. Dichas redes tienen un gran uso en el campo de la Programación Neurolingüística (PNL), como descifrar el sentimiento de una oración, el sentimiento general no depende solo de los trabajos individuales, sino también del orden en que las palabras se agrupan sintácticamente en la oración [6].

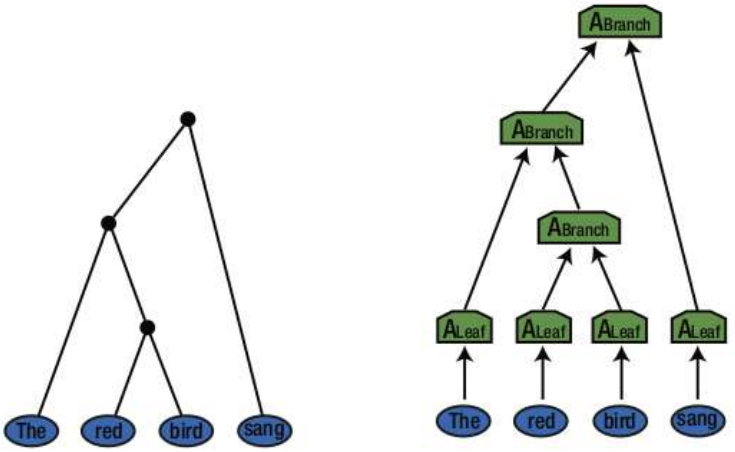


Figura 1.15: Red neuronal recursiva [6].

Perceptrón de una capa: Un perceptrón de una sola capa es un clasificador binario lineal simple. Toma entradas y pesos asociados y los combina para producir la salida que se utiliza para la clasificación. No tiene capas ocultas. La regresión logística es el perceptrón de una sola capa [5].

La Figura 1.16 muestra un modelo lineal con una entrada (X) y una salida (Y). El modelo de entrada única tiene un vector X con peso W y sesgo b. La salida, Y, es WX + b, que es el modelo lineal [5].

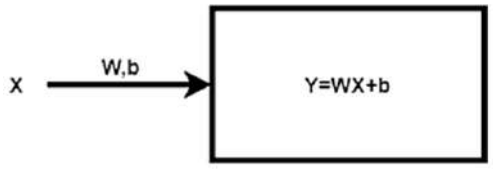


Figura 1.16: Vector de entrada única [5]

Perceptrón Multicapa

El perceptrón multicapa (MLP) es una red neuronal artificial formada por múltiples capas, de tal manera que tiene capacidad para resolver problemas que no son linealmente separables y consta de al menos una capa oculta de nodos que no sea la capa de entrada y la capa de salida como se muestra en la Figura 1.17, cada nodo de una capa que no sea la capa de entrada se denomina neurona que utiliza una función de activación no lineal como sigmoid o ReLU. Un MLP utiliza una técnica de aprendizaje supervisado llamada retropropagación para el entrenamiento, mientras que minimiza la función de pérdida como la entropía cruzada. Utiliza un optimizador para ajustar los parámetros (peso y sesgo). Sus múltiples capas y activación no lineal distinguen un MLP de un perceptrón lineal. Un perceptrón multicapa es una forma básica de una red neuronal profunda [5].

Los perceptrones multicapa (MLP) pertenecen a la categoría de redes neuronales de avance y normalmente tienen las siguientes propiedades:

• Capas ocultas con cualquier cantidad de neuronas.

• Una capa de entrada que utiliza funciones lineales.

• Capa (s) oculta (s) que utilizan una función de activación, como sigmoide.

• Una función de activación que proporciona cualquier número de salidas.

• Conexiones establecidas correctamente entre la capa de entrada, las capas ocultas y la capa de salida [6].

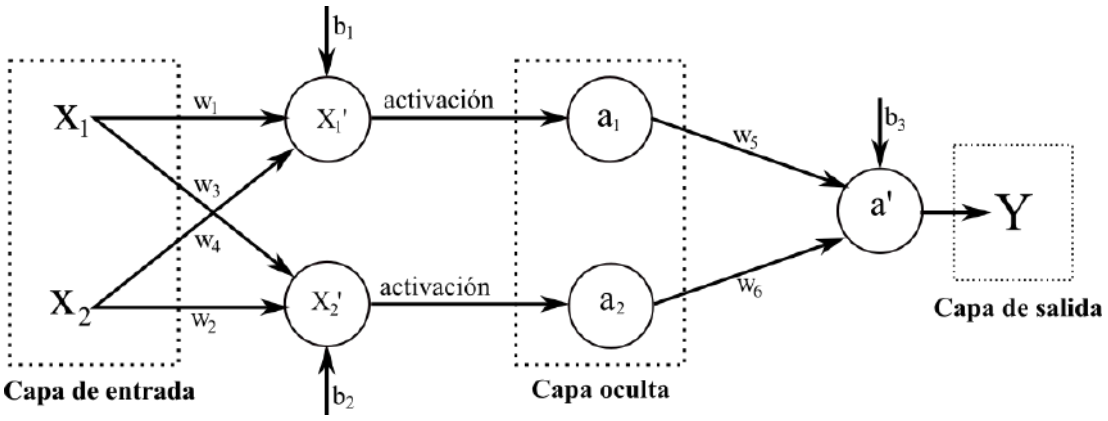


Figura 1.17: Diagrama de flujo para perceptrón multicapa [5].

Propagación hacia atrás

La propagación hacia atrás o *backpropagation* es un algoritmo que funciona mediante la determinación de la pérdida (o error) en la salida y luego propagándolo de nuevo hacia atrás en la red, con el objetivo de modificar los pesos de una red neuronal para minimizar el error de las salidas de la red en comparación con alguna salida esperada en respuesta a las entradas correspondientes como se muestra en la Figura 1.18 es un algoritmo de aprendizaje supervisado que permite corregir la red con respecto a los errores específicos cometidos [6].

El algoritmo general es el siguiente:

1. Presenta un patrón de entrada de entrenamiento y propague a través de la red para obtener una salida.
2. Compara las salidas predichas con las salidas esperadas y calcula el error.

3. Calcula las derivadas del error con respecto a los pesos de la red.

4. Ajusta los pesos para minimizar el error.

5. Reitera el proceso hasta obtener la salida deseada.

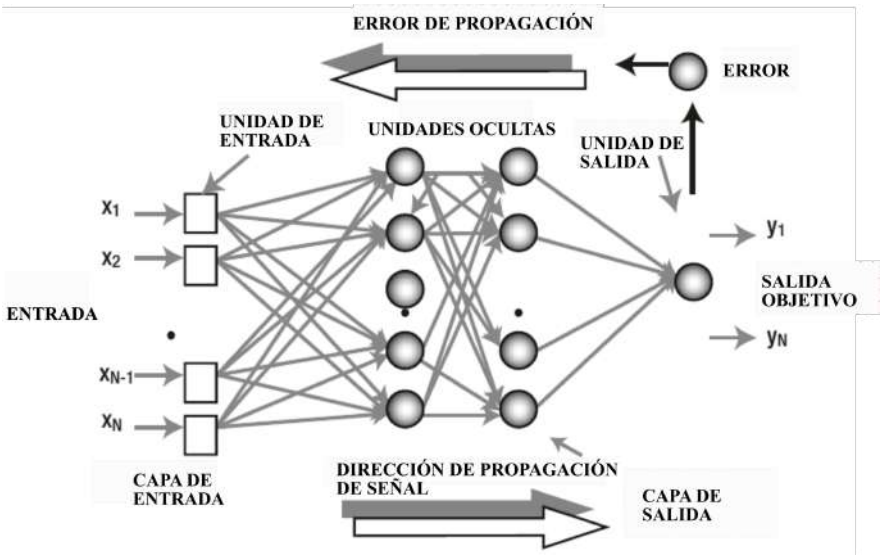


Figura 1.18: Mecanismo de retropropagación en un ANN

Inicialmente, todos los pesos de borde se asignan aleatoriamente. Para cada entrada en el conjunto de datos de entrenamiento, el ANN se activa y se observa su salida. Esta salida se compara con la salida deseada que ya se conoce, y el error se "propaga"de nuevo a la capa anterior. Este error se observa y los pesos se “ajustan” en consecuencia. Este proceso se repite hasta que el error de salida esté por debajo de un umbral predeterminado. Una vez que termina el algoritmo anterior, se tiene una ANN “aprendido”, que se considera listo para trabajar con “nuevas” entradas [6].

1.3.14 Funciones de pérdida y optimizadores

La función de perdida y la optimización es un método para evaluar qué tan bien algoritmo específico modela los datos dados, con el objetivo de controlar la salida de una red neuronal, para poder medir qué tan lejos está esta salida de lo que esperaba. En la Figura 1.19 se muestra como la función de pérdida toma las predicciones de la red y el objetivo verdadero (lo que quería que la red produjera) y calcula un puntaje de distancia, capturando qué tan bien la red ha funcionado en el modelo de la red,

utilizar esta puntuación como señal de retroalimentación para ajustar el valor de los pesos, en una dirección que disminuya la puntuación de pérdida es el trabajo principal del optimizador, que implementa lo que se llama el algoritmo de *Backpropagation*: el algoritmo central en el aprendizaje profundo [3].

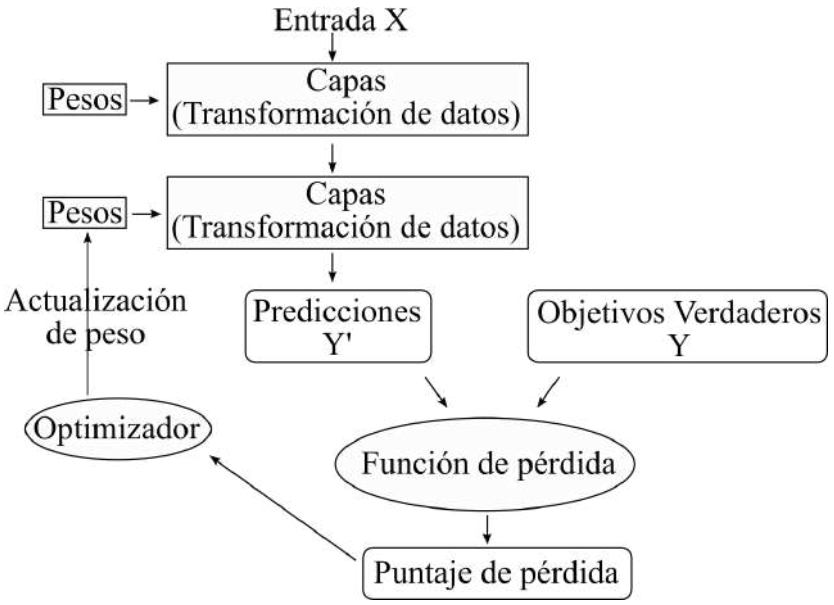


Figura 1.19: Función de pérdida con retroalimentación para ajustar los pesos [3].

Inicialmente, los pesos de la red se les asignan valores aleatorios, por lo que la red implementa una serie de transformaciones aleatorias. Naturalmente, su rendimiento está lejos de lo que debería ser idealmente, y el puntaje de pérdida es muy alto. Pero con cada ejemplo que procesa la red, los pesos se ajustan un poco en la dirección correcta y el puntaje de pérdida disminuye. Este es el ciclo de entrenamiento, que, repetido un número suficiente de veces (generalmente decenas de iteraciones en miles de ejemplos), produce valores de peso que minimizan la función de pérdida. Una red con una pérdida mínima es aquella para la cual las salidas están lo más cerca posible de los objetivos [3].

1.3.15 Librerías de Python para aprendizaje profundo.

Python es un lenguaje de programación multiparadigma, ya que soporta orientación a objetos, programación imperativa y en menor medida, programación funcional. Existen varias librerías para el aprendizaje profundo desarrollado en Python el cual no

es necesario conocer todas las complejidades del lenguaje para aplicarlo a “Machine Learning”, las librerías del aprendizaje profundo de Python que se consideran los más útiles y populares al desarrollo del aprendizaje automático y aprendizaje profundo, se describen a continuación:

Keras

Keras es una biblioteca de redes neuronales altamente modular, que se ejecuta sobre Theano o TensorFlow. Keras es una de las bibliotecas que admite tanto CNN (Convolutional Neural Network) como RNN (Red Neuronal Recurrente), y funciona sin esfuerzo en GPU y CPU. Un modelo se entiende como una secuencia o un gráfico de módulos independientes totalmente configurables que se pueden conectar con la menor cantidad de restricciones posible. En particular, las capas neurales, las funciones de costo, los optimizadores, los esquemas de inicialización, las funciones de activación, los esquemas de regularización son módulos independientes que se pueden combinar para crear nuevos modelos [6].

Tensorflow

Tensorflow es una biblioteca de software de código abierto para *Machine Learning* lanzada por Google en noviembre de 2015. Tensorflow se basa en el sistema interno que Google utiliza para impulsar sus sistemas de investigación y producción. Tensorflow es bastante similar a Theano y puede considerarse como el intento de Google de proporcionar una actualización a Theano al proporcionar interfaces fáciles de usar en Aprendizaje profundo, redes neuronales y Aprendizaje automático con un fuerte enfoque en prototipos rápidos y construcciones de implementación de modelos. Al igual que Theano, también proporciona construcciones para las matemáticas simbólicas, que luego se traducen en gráficos computacionales. Estos gráficos se compilan en código de nivel inferior y se ejecutan de manera eficiente. Al igual que theano, tensorflow también admite CPU y GPU sin problemas. De hecho, tensorflow funciona mejor en un TPU, conocido como la Unidad de procesamiento de tensor, que fue inventada por Google. Además de tener una API de Python, el tensorflow también está expuesto por las API a los lenguajes C ++, Haskell, Java y Go. Una

de las principales diferencias que tiene tensorflow en comparación con theano es el soporte para operaciones de alto nivel, que facilitan el proceso de Machine Learning y su enfoque en el desarrollo del modelo, así como el despliegue en la producción y el servicio del modelo a través de múltiples mecanismos [4].

NumPy

Se usa particularmente para la computación científica en Python. Está diseñado para manipular eficientemente grandes matrices multidimensionales de registros arbitrarios, sin sacrificar demasiada velocidad por pequeñas matrices multidimensionales. También podría usarse como un contenedor multidimensional para datos genéricos. La capacidad de NumPy para crear matrices de tipo arbitrario, que también hace que NumPy sea adecuado para interactuar con aplicaciones de base de datos de uso general, lo convierte en una de las bibliotecas más útiles [6].

Scikit-learn

Es una de las bibliotecas de "Machine Learning", más populares de la actualidad, usa numpy ampliamente para operaciones de matriz y álgebra lineal de alto rendimiento, es compatible con la mayoría de los algoritmos clásicos de aprendizaje supervisados y no supervisados [26].

Pandas

Es una biblioteca de software de código abierto. DataFrames y Series son dos de sus principales estructuras de datos que se utilizan ampliamente para fines de análisis de datos. Series es una matriz indexada unidimensional, y DataFrame es una estructura de datos tabular con índices de nivel de columna y fila. Pandas es una gran herramienta para preprocesar conjuntos de datos y ofrece un rendimiento altamente optimizado [6].

Matplotlib

Es una herramienta estándar que proporciona la capacidad de generar diagramas, histogramas, espectros de potencia, gráficos de barras, diagramas de error, diagramas

de dispersión, etc., que se pueden incrustar fácilmente en Jupyter Notebook para visualizar los datos y resultados obtenidos de los modelos [6].

Jupyter Notebook:

Es una aplicación web de código abierto que permite crear documentos que contienen código en vivo, ecuaciones, visualizaciones y texto narrativo, además de ser una herramienta simple y poderosa para manipular problemas de análisis de datos en código Python. La forma preferida de realizar experimentos de aprendizaje profundo Jupyter Notebook es una excelente manera de realizar experimentos de datos, aprendizaje automático y aprendizaje profundo [3].

1.3.16 Comunicación inalámbrica

La comunicación inalámbrica es la transferencia de información o energía entre dos o más puntos que no están conectados por un conductor eléctrico, se propagan en banda libre o privada para transmitir, es decir, sin la necesidad de cables para la conexión entre el emisor y el receptor [27].

Redes inalámbricas

La red inalámbrica es un término que se utiliza en informática para designar la conexión de nodos sin necesidad de una conexión física, ésta se da por medio de ondas electromagnéticas, la transmisión y la recepción se realizan a través de puertos [28].

1.3.17 Tecnología de transmisión

La tecnología de transmisión es la transferencia física de datos, por un canal de comunicación. Las redes inalámbricas se pueden implementar con diversas tecnologías, las mismas que se encuentran ligadas a estándares que permite y regulan su utilización [29]. Entre las tecnologías de transmisión las más utilizadas son las siguientes:

Bluetooth

El Bluetooth es un protocolo de comunicaciones que sirve para la transmisión inalámbrica de datos entre diferentes dispositivos y operan en la banda ISM (industrial, scientific, medical) de los 2.4 GHz, utiliza la modulación GFSK (Gaussian frequency shift keying) con un tiempo de ancho de banda (BT) = 0.5. Para evitar la interferencia y la perdida de información Bluetooth utiliza un transmisor-receptor de frequency hop (salto de frecuencia), proporcionan conexiones punto a punto, o una conexión punto a multipunto [30].

ZigBee

ZigBee es el nombre de la especificación de un conjunto de protocolos de alto nivel de comunicación inalámbrica para su utilización con radiodifusión digital de bajo consumo, basada en el estándar IEEE 802.15.4 de redes inalámbricas de área personal (wireless personal area network, WPAN). IEE.802.15.4 define las capas física y MAC, y Zigbee define las capas de red y aplicación. Su objetivo son las aplicaciones que requieren comunicaciones seguras con baja tasa de envío de datos y maximización de la vida útil de sus baterías [31].

Zigbee soporta topologıas comunes en redes como topología estrella y árbol, aunque cabe destacar que estas últimas topologías deben ser implementadas manualmente por el usuario, soporta por defecto una topologia de malla o Mesh la cual entrega las ventajas de ser calculada de forma algoritmica al momento de inicializar la red y de su capacidad auto regenerativa si es que un dispositivo enrutador sale de la red [31].

WiFi

WI-FI es una tecnología inalámbrica basada en el estándar IEEE 802.11, este estándar define y gobierna las redes de área local inalámbricas (WLAN) que operan en el espectro de los 2,4 GHz. El estándar 802.11 es una familia de especificaciones, entre las cuales se destacan las descritas en la Tabla 1.2 [32].

Tabla 1.2: Principales Estándares de la Familia 802.11.

	802.11b	**802.11a**	**802.11g**	**802.11b+**
Velocidad anunciada	11Mbps	54Mbps	54Mbps	22Mbps
Velocidad media obtenida	4-5Mbps	27Mbps	25Mbps	6Mbps
Frecuencia	2,4GHz	5GHz	2,4GHz	2,4GHz
Modulación	DSSS/CCK	OFDM	DSSS/PBCC	PBCC
Canales	11	12	11	11

Elaborado por: El investigador, en base a [32]

GSM Global System Mobile Comunications

La red GSM nace como un estándar internacional de comunicaciones digitales móviles y se caracteriza por tener una comunicación digital por voz y soporte en servicios de datos. La interfaz de la red se basa en el acceso múltiple por división de tiempo TDMA, logrando obtener un mayor radio de banda que es compartida por múltiples usuarios alojando una o más compuertas de tiempos para cada subscriptor. Con GSM la transferencia de datos es llevada sobre conexiones de circuitos conmutados [33].

GPRS General Packet Radio Service

General Packet Radio Service (GPRS) es una tecnología descendiente de GSM que utiliza la conmutación de paquetes, paralelo a la conmutación de circuitos. El sistema GPRS actualiza los servicios de datos GSM para hacerlos compatibles con LANs ,WANs e Internet. Mientras el actual sistema GSM fue originariamente diseñado con un especial énfasis en las sesiones de voz, el principal objetivo de GPRS es ofrecer un acceso a redes de datos estándar, como TCT/IP. Estas redes consideran GPRS como una subred normal. El actual sistema GSM opera en un modo de transmisión de circuitos conmutados .^{ex}tremo a extremo", en el cual los circuitos son reservados a lo largo del sistema para el uso de una sola comunicación incluso cuando no se transmiten datos [34].

Aplicaciones

Las aplicaciones de GPRS son las siguientes:

- Permite el envío y la recepción de información a los celulares dividiendo la información en paquetes.
- GPRS puede enviar y recibir información (e-mails, imágenes, gráficos, etc.) utilizando el mismo equipo celular a través del navegador WAP.
- La red GPRS habilita la utilización de más recursos de radio para ser alojados en conexiones de paquetes que en circuitos conmutados como se hace en GSM [34].

1.3.18 Microcontrolador

Un microcontrolador es un dispositivo electrónico capaz de llevar a cabo procesos lógicos para desempeñar una tarea específica.Dicha tarea debe ser programada por el usuario a través de un lenguaje de programación. Dentro de los microcontroladores existen dos arquitecturas básicas de hardware Von Neumann y Harvard [35].

Von Neumann: Se caracteriza por tener una memoria única para los datos y las instrucciones del programa. A dicha memoria se accede a través de un sistema de buses único (control, direcciones y datos), en la Figura 1.20 se muestra la arquitectura del esquema Von Neumann [35].

Figura 1.20: Esquema de la arquitectura Von Neumann.

Harvard: Este modelo tiene la unidad central de proceso (CPU) conectada a dos memorias, una con las instrucciones y otra con los datos, por medio de dos buses

diferentes. Una de las memorias contiene solamente las instrucciones del programa (Memoria de Programa), y la otra sólo almacena datos (Memoria de Datos), en la Figura 1.21 se aprecia la arquitectura Harvard básicas de hardware [35].

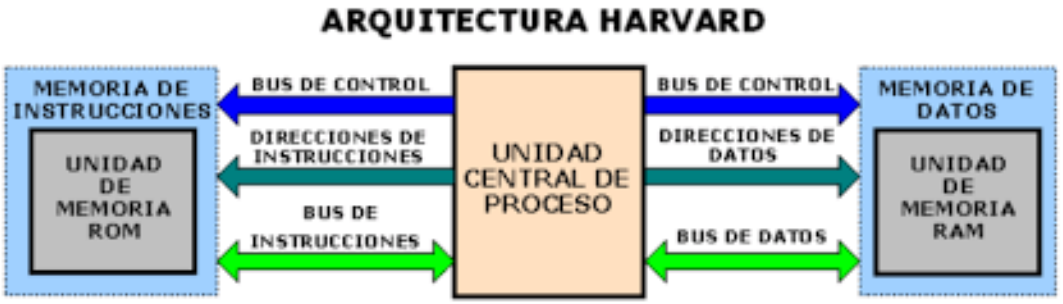


Figura 1.21: Esquema de la arquitectura Harvard.

1.3.19 Sensores

Los sensores son piezas clave en los proyectos con microcontroladores, permiten cuantificar una señal externa y enviarla al microcontrolador. Además, existen diferentes familias de sensores y cada uno de ellos tiene un uso concreto [36].

Sensor de temperatura

Los sensores de temperatura convierten una magnitud física en una resistencia o tensión eléctrica. El uso de sensores de temperatura es muy amplio. Sea que se trate de la temperatura ambiental en la casa o en la oficina o la temperatura precisa de un material en proceso de ebullición, la medición de temperatura en el ámbito privado o industrial es muy importante. Los sensores de temperatura usan diferentes efectos físicos para convertir la temperatura en una magnitud eléctrica [37].

Sensor de humedad

Los sensores de humedad convierten la magnitud física de humedad del aire en una señal normalizada y tiene un amplio ámbito de aplicación y sirve para determinar la humedad relativa [%] y absoluta [g/m³] con una gran precisión [37].

Sensor de calidad de aire

Los sensores de calidad de aire son dispositivos usados para la detección de contaminantes en el aire. Esto incluye partículas, contaminantes y gases nocivos que pueden ser perjudiciales para la salud humana. Se utilizan en aplicaciones como el

monitoreo de la calidad del aire, la detección de gas en industrias, controladores de combustión y generadores de oxígeno en aviones [37].

Sensores mecánicos

Son dispositivos que tienden a cambiar su comportamiento bajo la acción de una magnitud física que pueden de manera directa o indirectamente enviar/transmitir una señal que indica el cambio. Son utilizados para medir desplazamiento, posición, tensión, movimiento, presión, fuljo [37].

Sensor ultrasónico

Este sensor emite cíclicamente un impulso acústico de alta frecuencia y corta duración. Este impulso se propaga a la velocidad del sonido por el aire. Al encontrar un objeto, es reflejado y vuelve como eco al sensor ultrasónico. Este último calcula internamente la distancia hacia el objeto, basado en el tiempo transcurrido entre la emisión de la señal acústica y la recepción de la señal de eco [37].

Sensor fotoeléctrico

Son dispositivos que detecta la presencia o alguna característica en particular de un objeto mediante luz visible o no visible Se pueden aplicar para detectar presencia, tamaño, color y brillo de objetos. Existen tres modos de configuración fotoeléctrica [38].

1. Modelo reflectivo: El modelo reflectivo de la Figura 1.22 el emisor de luz y los elementos receptores están contenidos en una sola carcasa. El sensor recibe la luz reflejada desde el objeto [38] .

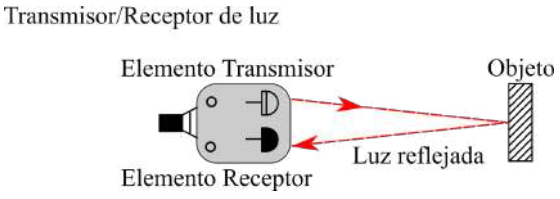


Figura 1.22: Modelo reflectivo.
Elaborado por: El investigador, en base a [38].

2. Modelo de barrera: El modelo de barrera de la Figura 1.23 el transmisor y el

receptor están separados, cuando el objeto se encuentra entre el transmisor y el receptor, se interrumpe la luz y manda una alerta [38].

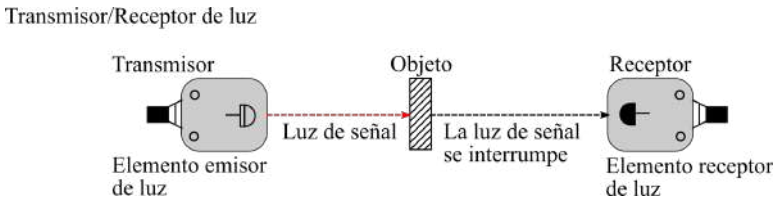


Figura 1.23: Modelo de barrera.
Elaborado por: El investigador, en base a [38]

3. Modelo retroreflectivo: El modelo retroreflectivo de la Figura 1.24 el emisor de luz y los elementos receptores están contenidas en un mismo recinto. La luz del elemento emisor incide en el reflector y regresa al elemento receptor de luz. Cuando hay un objeto presente, se interrumpe la luz [38].

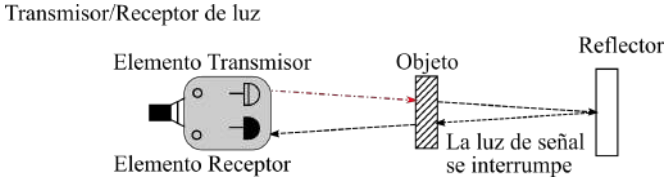


Figura 1.24: Modelo retroreflectivo.
Elaborado por: El investigador, en base a [38].

Sensores capacitivos:

Debido a su naturaleza no direccional, el sensor capacitivo es capas de medir cierta capacitancia de objetos en el entorno que están siempre presentes, utiliza la propiedad eléctrica de la capacitancia y el cambio de capacitancia basado en un cambio en el campo eléctrico alrededor de la cara activa del sensor. Las aplicaciones mas comunes incluyen presión, humedad , nivel de fluido, fuerza, posición y desplazamiento [37].

1.4 Objetivos

1.4.1 Objetivo general

Implementar un sistema de monitoreo apícola mediante el uso de redes neuronales artificiales para identificar la variación de población.

1.4.2 Objetivos específicos

• Identificar las variables críticas y sus rangos óptimos de acuerdo con los requerimientos establecidos para la apicultura.
• Aplicar redes neuronales de predicción de variación de población apícola.
• Desarrollar una aplicación para visualizar la información del dispositivo de monitoreo apícola inalámbricamente.

Capítulo II

METODOLOGÍA

2.1 Materiales

Para el desarrollo de la metodología para el diseño del sistema de monitoreo apícola para el presente proyecto de investigación, se requiere de los siguientes materiales: Artículos de revistas, Tesis, Libros, y demás documentos relacionados al tema principal, Además de, registros de datos de información relevantes en el área de la apicultura para mantener la estabilidad de una colmena de abejas melíferas.

2.2 Métodos

2.2.1 Modalidad de la Investigación

La modalidad del proyecto para la construcción de un prototipo que registra las actividades apícolas es de Investigación y Desarrollo, la cual se realizo mediante las siguientes técnicas:

Se utilizo la investigación aplicada, ya que se puso en práctica los conocimientos científicos adquiridos como también la información sobre la tecnología actual relacionada al tema, que sirven para dar solución al problema planteado.

Se enmarca dentro de la investigación bibliográfica debido a que el sustento científico del tema planteado se obtuvo de libros, publicaciones, artículos científicos y repositorios disponibles en el internet.

Se utilizo la investigación de campo porque se trata de un estudio de los hechos, en

el lugar que se produce para ello se utilizo técnicas como la observación, la entrevista que permite estar en contacto directo al investigador con la finalidad de recolectar y registrar información.

2.2.2 Recolección de Información

Se recolecto información en base al uso de documentos, revistas, libros, proyectos desarrollados, por lo que se tomo en cuenta bases de datos confiables de información, cada uno de estos están relacionados y vinculados al proceso de creación de redes neuronales artificiales que permiten desarrollar sistemas autónomos vinculados a la apicultura, por lo que se requiere de la investigación bibliográfica.

2.2.3 Procesamiento y Análisis de Datos

Se obtuvo información apropiada para la investigación planteada, el cual se formo parte de un proceso analítico, el cual consiste en:

• Revisión de la información recogida.
• Revisión del estado actual sobre la monitorización de la apicultura basados en redes neuronales.
• Análisis de la información adquirida para plantear estrategias que permitan a alcanzar una solución óptima al problema planteado.
• Análisis e interpretación de los Resultados.

El análisis de resultados del procesamiento de la información está ligado a los conceptos del marco teórico relacionado con los objetivos de la investigación propuesta para posteriormente exponer las conclusiones y recomendaciones del presente proyecto de investigación, además de fuentes bibliográficas relacionadas al entrenamiento de redes neuronales artificiales aplicados a la apicultura, como papers, tesis de grado, páginas web, libros, folletos y revistas científicas.

2.2.4 Desarrollo del Proyecto

1. Recolección de información de las variables y condiciones óptimas para el desarrollo del sistema de monitoreo apícola
2. Selección de los parámetros necesarios para la implementación del sistema de monitoreo apícola.
3. Investigar los tipos de redes neuronales artificiales adecuados para el sistema de monitoreo.
4. Selección del modelo de red apropiado para la etapa de entrenamiento de la red neuronal.
5. Analizar y evaluar el modelo propuesto de la red neural para determinar la variación de población apícola.
6. Ejecutar las predicciones correspondientes del conjunto de datos de prueba para conocer la precisión del modelo.
7. Selección del dispositivo de comunicación inalámbrica adecuada a emplearse en el sistema.
8. Implementación del prototipo sistema electrónico de comunicación para las actividades apícolas.
9. Desarrollo de la aplicación de monitorización apícola a nivel de hardware y software.
10. Adaptación de la etapa de monitorización apícola con la etapa de visualización y control.
11. Pruebas y corrección de errores del prototipo del sistema electrónico de comunicación planteado
12. Elaboración del trabajo escrito.

Capítulo III

RESULTADOS Y DISCUSIÓN

3.1 Análisis y discusión de los resultados

La implementación de un prototipo para monitorizar las actividades apícolas, capaz de registrar los problemas y condiciones que afecta a la disminución de población apícola a través de análisis de la arquitectura de modelos basado en redes neuronales artificiales que permitirá a los apicultores identificar posibles problemas que causan la variación de población de una colmena de abejas.

3.2 Desarrollo de la propuesta

El presente proyecto consiste en la construcción de un prototipo de sistema de monitoreo apícola, con tecnología Wi-Fi para identificar la variación de población mediante el uso de redes neuronales artificiales. El sistema permite supervisar el estado general de una colmena de abejas, gracias a los elementos de medida se puede identificar las variable físicas que afectan directamente la salud y la producción de población, la lectura de dichas variables son enviadas por medio de tecnología inalámbrica a una base de datos almacenada en un servidor web.

Los datos registrados en el sistema de monitoreo son usados como datos de entrenamiento para predecir la variación de población de una colmena, a demás de predecir las condiciones de temperatura, humedad, calidad de aire y peso de la colmena, aplicando técnicas de aprendizaje automático basados en redes neuronales artificiales, permitiendo al apicultor tomar medidas preventivas ante situaciones de riesgo para la colmena.

El diseño del sistema se basó según los requerimientos y condiciones óptimas de las variables físicas dentro de una colmena de abejas melíferas establecidos para la apicultura y tiene como objetivo principal predecir la variación de población de una colmena construido con tecnología accesible que utilice Redes Neuronales Artificiales (RNA). Con lo cual la presente investigación se enfoca en obtener información relevante de las condiciones dentro de la colmena, y utilizarlo para informar al apicultor con el objetivo de mantener una colmena saludable.

3.3 Elementos que conforman el sistema de monitoreo apícola.

La arquitectura general del sistema de monitoreo apícola se presenta en el diagrama de la Figura 3.1, en el cual se muestra los elementos que conforman y caracterizan el proceso de desarrollo del diseño del sistema de monitoreo apicola.

Figura 3.1: Elementos del sistema de monitoreo apícola.
Elaborado por: El investigador, en base a [39].

3.4 Variables y rangos de interés dentro de una colmena.

Para el desarrollo del sistema de monitoreo apícola, primero se debe identificar las variables físicas y sus rangos óptimos dentro de una colmena, para lo cual se recopilo información relacionada y vinculada a las actividades de la apicultura. Los parámetros que sostienen la estabilidad en la colmena, que pueden indicar el estado en el que se encuentra la colonia de abejas melíferas se identificaron como: temperatura, humedad, niveles de dióxido de carbono (CO2) y oxígeno, que están estrechamente relacionados con la salud y la productividad individual de la colmena.

3.4.1 Temperatura

La temperatura del nido de cría es de extrema importancia para la colonia de abejas melíferas que necesitan mantener la temperatura de nido entre 32°C y 36°C, y

óptimamente a 35°C, para que la cría se desarrolle normalmente, las abejas obreras regulan la temperatura al expulsar el aire caliente del nido cuando se percibe que la temperatura es demasiado alta y se agrupan y generan calor metabólico cuando se percibe que la temperatura es demasiado baja, incluso pequeñas desviaciones de ± 0.5°C de las temperaturas óptimas de cría tienen una influencia significativa en el desarrollo de la cría y la salud de las abejas adultas resultantes [40], [41], [42].

3.4.2 Humedad relativa

La humedad relativa es la cantidad de agua retenida en el aire en relación con la cantidad máxima de agua que se puede retener en el aire a una temperatura dada. Cuanto más cálido es el aire, más agua puede contener, por lo tanto, a medida que fluctúa la temperatura, también lo hace la humedad relativa.

Durante el período de cría, la mediana de los niveles de humedad en el nido de una colonia sana y fuerte es de entre 50 % y 60 %. Raramente se encuentra por debajo del 40 % y por encima del 80 %, los niveles de humedad altos o bajos afectan la salud de las crías y las abejas adultas, al elevar la humedad del 68 % al 87 % aumenta el porcentaje de momificación de la cría causada por la cría de tiza conocido como una enfermedad infectocontagiosa de origen fungal en un 8 %. La tasa reproductiva de Varoa la cual es un género de ácaros que produce la enfermedad denominada varroasis o varroosis, disminuye con el aumento de la humedad. La termorregulación y la concentración de néctar también están estrechamente relacionadas con los niveles de humedad en la colmena [43].

3.4.3 Dióxido de carbono.

Las abejas acumulan dióxido de carbono dentro de sus colonias cuando se encuentran en un espacio confinado donde la ventilación es restringida. Dejar que los niveles de CO2 se acumulen demasiado puede matar a las abejas. Aún se investiga un punto óptimo de CO2 que no sea nocivo para las abejas, sino para los ácaros que afectan a la productividad de la colmena. Otros comportamientos o fisiologías de las abejas melíferas afectadas por cambios en la concentración de CO2 incluyen el desarrollo de glándulas hipofaríngeas, cuerpo graso, glándulas de cera, ovarios, cuerpos allata,

química cerebral, recolección y consumo de polen, polietismo, vida útil y producción de huevos en abejas obreras [44], [45].

3.4.4 Densidad poblacional

La densidad poblacional de abejas apis mellifera es el número promedio de habitantes que residen en una colmena, la regla de Farrar conocida por los apicultores hace muchos años, dice que cuanto mayor es la población de una colmena, mayor es la producción individual de cada abeja. Esto equivale a decir que aumenta la productividad y se conoce como un principio de sinergia. Esto se debe a que al aumentar el número de abejas de una colmena, también aumenta la proporción de pecoreadoras como se muestra en la Tabla 3.1.

Tabla 3.1: Regla de Farrar en la apicultura.

Total de Obreras	**10.000**	**20.000**	**30.000**	**40.000**	**50.000**	**60.000**
Pecoreadoras	2.000	5.000	10.000	20.000	30.000	39.000
Porcentaje obreras	20 %	25 %	30 %	50 %	60 %	65 %
Peso de la población	1Kg	2Kg	3Kg	4Kg	5Kg	6Kg
Rendimiento miel	1Kg	4Kg	9Kg	16Kg	25Kg	36Kg

Elaborado por: El investigador, en base a [46].

También se puede hacer un cálculo matemático por el cual conociendo la población de abejas de una colmena, puede estimarse la producción de esta aproximadamente, se dice que la capacidad de producción es igual al cuadrado del peso de la población [46], [47].

3.4.5 Peso de la colmena

Los apicultores y los investigadores de las abejas pesan las colmenas diariamente o semanalmente para ayudar a determinar el mejor momento para cosechar miel o estimar las reservas de alimentos para períodos sin flujo de néctar. El pesaje es rápido, requiere poco entrenamiento y no es perjudicial para la colonia, por lo que se puede hacer en cualquier época del año. Los datos sobre los cambios a lo largo del tiempo

genera datos sobre el crecimiento y la actividad de la colmena, además proporciona información precisa sobre el momento y el tamaño de los eventos, como enjambres y fenómenos imprevistos de la colmena, y permite a los investigadores controlar los cambios de peso debido a la alimentación [48].

3.5 Análisis de los componentes electrónicos requeridos para el sistema de monitoreo apícola.

3.5.1 Selección del sensor de temperatura y humedad relativa.

Para el sistema de monitoreo apícola, es necesario medir la temperatura y la humedad dentro de la colmena de abejas, para lo cual se hace un análisis de los diferentes tipos de sensores en el que se muestra en la Tabla 3.2 para tal acción:

Tabla 3.2: Comparación de las características de los sensor de temperatura/humedad.

PARÁMETROS	**SENSOR DHT21**	**SENSOR DHT11**	**SENSOR MAX6675**
Voltaje de Operación:	3V - 5V DC	3.5V - 5.5V DC	5V
Corriente de trabajo:	2.5 mA	1mA - 1.5mA	50mA
Rango de medición de temperatura:	0 a 50 °C	-40 hasta 80°C	0° - 1023°C
Precisión de medición de temperatura:	±2.0 °C	+- 0.5°C	0 ° - 700 °C.
Resolución Temperatura:	0.1°C	0.1°C	0.25ºC
Rango de medición de humedad:	20 % a 90 % RH	0 a 100 % RH	————
Precisión de medición de humedad:	5 % RH	+- 3 %	————
Resolución Humedad:	1 % RH	0.1 %RH	————
Tiempo de sensado:	1 seg	2 seg	0.17 - 0.22 seg

Elaborado por: El investigador, en base a [49]

Para la selección del sensor se opto por utilizar dos sensores el DHT21 como sensor de humedad por su rango de medición de 0 a 100 %, su precisión de +-3 % y a su bajo consumo de corriente de 1mA a 1.5mA, y el sensor MAX6675 como sensor de temperatura, debido a su rango de precisión de medición de 0° a 700°C, ideales para satisfacer las magnitudes de las variables físicas de temperatura y humedad dentro de la colmena, en la Figura 3.2 se muestra la forma física del sensor.

(a) Sensor de humedad DHT21 (b) Sensor de temperatura MAX6675

Figura 3.2: Sensor de temperatura y humedad [49].

3.5.2 Selección del sensor de dióxido de carbono

Los niveles de dióxido de carbono dentro de la colmena es incierto ya que no se ha comprobado cual es el punto óptimo de CO2 que no sea nocivo para las abejas, por lo cual es necesario realizar un registro de los niveles de CO2 dentro de la colmena, en la Tabla 3.3 se analizan tres tipos de sensores de calidad de aire.

Tabla 3.3: Comparación de las características del los sensor de calidad de aire.

PARÁMETROS	**Sensor MQ135**	**Sensor CCS811**	**Grove - v1.3**
Concentración detectable	Amoniaco, sulfuro, benceno, humo, CO2	Dióxido de carbono y metales de oxido.	Monóxido de carbono, alcohol, acetona
Rango de detección	10ppm - 1000ppm	eCO2: 400 - 8192 , TVOC: 0 - 1187 ppm	10ppm - 1000ppm
Señal de salida	Analógica y salida de nivel TTL.	Interfaz digital estándar I2C	————
Consumo	800mW	————	Bajo consumo de energía.
Voltaje de Operación	5v	5v	3.3v - 5v.

Elaborado por: El investigador, en base a [7].

Los sensores de gas que se tomaron en consideración para la comparación tienen características comunes. Sin embargo, el sensor MQ135 tiene mejores parámetros para la detección de concentración de gas en diversos porcentajes, que los otros dos sensores, además es usado como control de calidad de aire y para la detección de contaminación en el medio ambiente, por lo cual es indispensable para mantener la estabilidad de la colmena en términos de salud dentro de la colmena, en la Figura 3.3 se muestra su forma física del sensor MQ135.

Figura 3.3: Sensor de calidad de aire MQ135 [7]

3.5.3 Construcción del sensor capacitivo para estimar la densidad poblacional de una colmena.

Para establecer la cantidad de población de abejas que salen y entran de la colmena, los apicultores usualmente modifican las colmenas, habilitando orificios que pasan a través de un área cerrada, como un túnel o tubería entre 7mm a 1cm de diámetro que únicamente son usados por las abejas como salida. En la Figura 3.4 se observa como rastrear el paso de las abejas por los orificios de salida de la colmena.

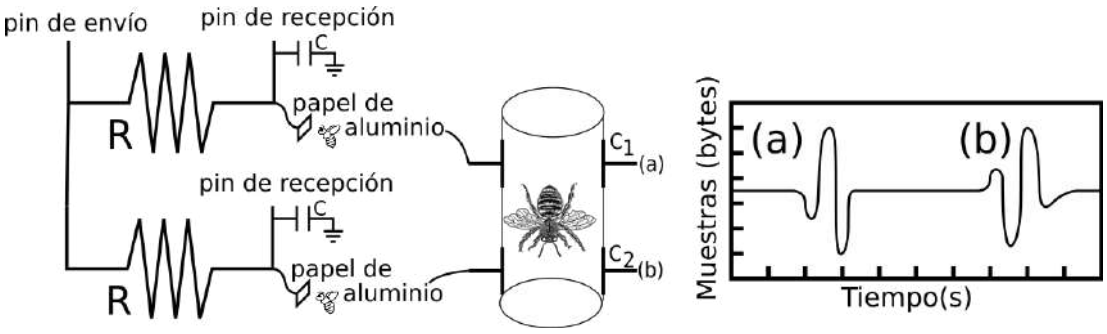
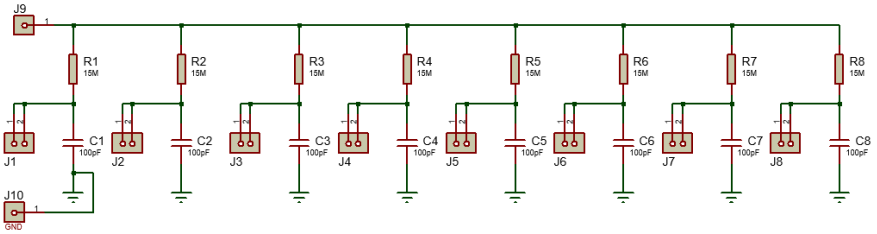


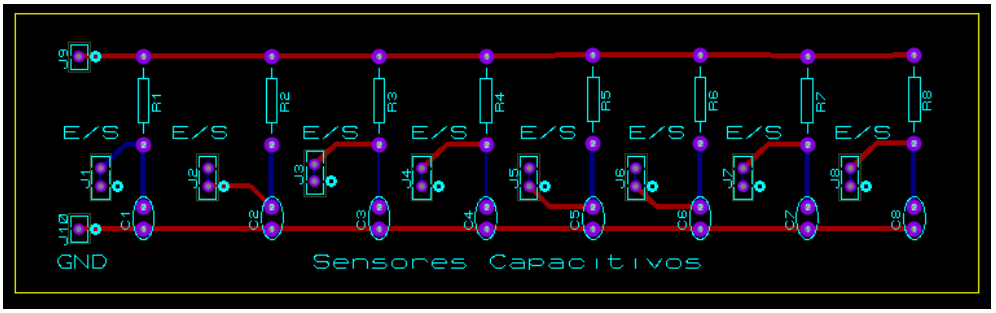
Figura 3.4: Funcionamiento del sensor capacitivo
Elaborado por: El investigador.

La configuración física incluye una resistencia de valor medio alto entre 100 kilohmios a 50 megaohmios entre el pin de envío y el pin de recepción del sensor. El pin de recepción es el terminal del sensor. Un cable conectado a este pin con un trozo de papel de aluminio en el extremo constituye un buen sensor de presencia a pocos milímetros de distancia dependiendo del valor de la resistencia entre el pin de envío y el pin de recepción, con una resistencia de 15 megaohmios se capturo la presencia entre 3 a 5 mm de distancia, se obtiene un rango de valores más útil si el sensor se cubre con papel, plástico u otro material aislante, de modo que las abejas no toquen realmente la lámina de aluminio, al conectar un condensador de 100 pF o más desde el pin del sensora GND del circuito mejora la estabilidad y la repetibilidad, en la Figura 3.5 se muestra el diseño de la placa electronica para los sensores capacitivos en el que consta de dieciséis resistencias (ocho de entrada y salida) de 15 megaohmios y 16 condensadores de 100pF

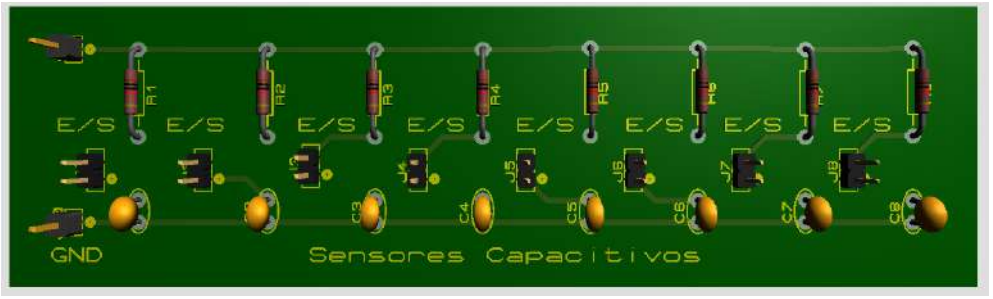
.



(a) Circuito esquemático.



(b) Diseño PCB de la placa electronica.



(c) Circuito 3D esquemático.

Figura 3.5: Diseño de la placa electrónica para los sensores capacitivos en el software Proteus.

Elaborado por: El investigador.

3.5.4 Sensor de carga.

Las celdas de carga son transductores de fuerza que se emplean para la comprobar o medir la fuerza, mediante la transformación de la magnitud mecánica en magnitud eléctrica, es decir, fuerza ejercida en voltaje, en la Tabla 3.4 se muestra el análisis comparativo de los de los sensores de presión o fuerza disponibles en el mercado.

Tabla 3.4: Sensor de peso para el prototipo del sistema de monitoreo apícola

Sensor de presión o fuerza	**SPRK-SEN10245**	**MF01**	**SHT-101**	**RESISTENCIA DE 5 KG**
Ventajas	Margen de error mínimo	-Bajo costo -Fácil uso -Variedad de aplicaciones	Margen mínimo de error -Alta sensibilidad	-Costo accesible -Alta precisión -Fácil uso
Desventajas	Requiere adaptación de señales	-Se obtiene rango de respuesta	-Precio elevado	- Requiere adaptación para la adquisición de los datos
Sensibilidad	1.0+-0.1	1.0+-0.1	2+-0.1	1,0 ± 0.15
Gama fuerza	40-50 kg	0-20 lb	20-40 kg	0-5 Kg

Elaborado por: El investigador, en base a [50], [51]

Los apicultores e investigadores de abejas pesan las colmenas diariamente o semanalmente por el cual se requiere una alta precisión de los datos, por ende el sensor que cumple con lo requerido es de resistencia de 5 kg por su precisión y bajo costo al igual que el rango de peso de 0-5 kg, ideal para estimar el peso de la colmena, para hacer uso de este sensor es necesario utilizar el transmisor HX711 como se muestra en la Figura 3.6 para obtener valores confiables y con buena precisión.

Figura 3.6: Celdas de carga de 5kg y su modulo HX711 [8].

3.5.5 Dispositivos de adquisición de datos.

La adquisición de datos consiste en tomar un conjunto de señales físicas, convertirlas en tensiones eléctricas y digitalizarlas de manera que se puedan ser procesadas por una computadora o un microcontrolador emitidos por los sensores previamente

seleccionados, el cual lo procesa para generar una trama que sea comprensible al momento de envío y recepción de los datos. En la Tabla 3.5 se analizaron tres placas electrónicas que existen el mercado, para adquirir las señales de las variables físicas dentro de la colmena.

Tabla 3.5: Análisis característico de microcontroladores.

PARÁMETROS	**ESP8266**	**PIC16F887**	**ATMEGA2560**
Voltaje de operación	3.3V	2V- 5.5 V	5 V
Voltaje de entrada	5V	5.5 V Max	7-12 V
Número de entradas digitales	9 pines GPIO	21	54
Número de entradas análogas	1	14	16
Frecuencia de operación	80MHz	0 - 20 Mhz	16 Mhz
Tipos de comunicación	I2C, SPI, WiFi 802.11 b/g/n	RS-485, RS-232 y LIN2.0	SERIE UART-I2C-SPI, RS232
Memoria interna	4MB de memoria FLASH (32 MBit)	8KB, 256 bytes de memoria EEPROM, 368 bytes de memoria RAM	256 KB, 8 KB de memoria SRAM y 4KB de EEPROM
Temperatura de operación	máxima de 125°	-45 – 150 ºC	-45 – 85 ºC

Elaborado por: El investigador, en base a [35]

Para el desarrollo del proyecto se optó por elegir el microcontrolador ATmega2560 el cual forma parte del módulo Arduino Mega, como el mas óptimo, debido a que posee un mayor numero de pines digitales y análogos, para el sistema de monitoreo se requiere un pin digital para el sensor de humedad, dos pines análogos para el sensor de temperatura, dos pines digitales para el sensor de carga, un pin análogo para el sensor de CO2, dieciocho pines digitales para los sensores capacitivos y dos pines digitales para el modulo ESP-01 para un total de 23 pines digitales y 3 pines análogos

suficientes para la implementación del sistema ya que cuenta con mayor numero de pines de entrada/salida para los sensores que se acoplan al microcontrolador, además de la memoria FLASH, SRAM, EEPROM son de mayor capacidad, lo suficientemente para la implementación del proyecto en la Figura 3.7 se muestra la forma física del microcontrolador ATmega2560.

Figura 3.7: Arduino mega 2560 [9].

3.5.6 Sistema de alimentación.

Las placas arduino son muy versátiles y admiten varias formas para el sistema de alimentación, el cual es muy fundamental para el desarrollo del proyecto debido a que es necesario contar con la potencia necesaria para operar de forma eficiente y evitar que se trasmitan datos erróneos, además de que las colmenas se encuentran en zonas fuera del alcance del suministro eléctrico. En la tabla 3.6 se muestra el análisis comparativo entre tres diferentes tipos de baterías.

Tabla 3.6: Comparación de las características del sistema de alimentación.

PARÁMETROS	**Batería Lipo**	**Baterías Li-on**	**Baterías Seca**
Modelo	903052	DP-9V USB650	6F22
Tipo	Li-polímero	Litio	Zinc-carbono,
Capacidad	1200 mAh	650 mAh	400mAh
Tensión Nominal	7,4 V	8 ,4 V	9 v
Voltaje de corte	–	7.1 V	–
Peso	0.02 kg	0.05 kg	0.04 kg
Dimensiones	18*30*54mm	48*26*16mm	24*15*46mm

Elaborado por: El investigador, en base a [10]

Para el sistema de alimentación se seleccionó la batería seca de zinc-carbono, en la Figura 3.8 se aprecia su forma fisica, dispone de gran cantidad de energía para la alimentación de la placa Arduino Mega 2560 y de los modulos: DTH21, MQ135 y HX711. Además de que el consumo de energético es mínimo, por lo que la batería abastece favorablemente la demanda de energía de los componentes mencionados.

Figura 3.8: Batería de Zinc 6F220 [10]

3.6 Análisis de los parámetros técnicos para el sistema de comunicación.

3.6.1 Selección de la tecnología inalámbrica.

Una vez analizado los componentes electrónicos del sistema de monitoreo apícola, se procedió a la selección de la tecnología adecuada para el diseño y construcción del prototipo, la utilización de la tecnología inalámbrica es necesaria debido a que proporciona la facilidad de envió de datos de las variables físicas dentro de la colmena. En la Tabla 3.7 se describe el análisis de diferentes tecnologías inalámbricas con sus respectivas características.

Tabla 3.7: Comparación de las tecnologías de comunicación inalámbricas.

Parámetros	Bluetooth	WI-FI	GSM-GPRS	ZIGBEE
Estandar	802.15.1	802.11	GSM	802.15.4
Frecuencia	2.4 GHz	2.4 – 5 Ghz	300-1200 Mhz	2.4GHz
Alcance	10 - 20 m	50-100 m	Depende de la operadora	1 - 75 m
Velocidad de transferencia de datos.	64Kbps	54Kbps	56 - 114 kbps	250Kbps
BW del canal	1 Mbps	2-100 Mbps	20 - 100 Mbps	20-250 Kps
Modulación	GFSK	PSK	GMSK	DSSS
Topología	Punto a punto, estrella, árbol	Bus, anillo, estrella, árbol	Topología celular	Punto a punto, estrella, malla, árbol, clúster

Elaborado por: El investigador, en base a [30], [32], [33], [31].

La implementación del proyecto se puede realizar con diversas tecnologías inalámbricas, sin embargo, se requiere analizar condiciones mínimas de las explotaciones apícolas, según la Legislación Consolidada establece el Real Decreto 209/2002 en el Articulo 8 se establece la ubicación del colmenar según los siguientes apartados:

1. Establecimientos colectivos de carácter público y centros urbanos, núcleos de población: 400 metros
2. Viviendas rurales habitadas e instalaciones pecuarias: 100 metros
3. Carreteras nacionales: 200 metros.
4. Carreteras comarcales: 50 metros
5. Caminos vecinales: 25 metros

No obstante, para las explotaciones de autoconsumo, otras distancias mínimas podrán ser establecidas por cada comunidad autónoma de acuerdo con las específicas características de la producción apícola en su ámbito territorial. La distancia

establecida para carreteras y caminos en el apartado 2 podrá reducirse en un 50 a 100m si el colmenar está en pendiente y a una altura o desnivel superior de dos metros con la horizontal de estas carreteras y caminos [52].

La ubicación del colmenar se localiza en las coordenadas 1°14’33.657"S y 78°37’49.599.º, en la Figura 3.9 se muestra la ubicación del apiario en una zona rural a 75 metros de la vivienda mas cercana cumpliendo los asentamientos apicolas establecidas por la Legislación Consolidada, por ende la tecnología inalámbrica a implementar en el prototipo de monitoreo apícola es WiFi el cual cumple con el alcance requerido para su implementación.

Figura 3.9: Zona de la ubicacion del colmenar
Elaborado por: El investigador

3.6.2 Selección del módulo Wi-Fi para el sistema de comunicación.

Para el sistema de comunicación es necesario determinar los parámetros técnico y económicos apropiados para el desarrollo del proyecto, en el apartado anterior se señaló la tecnología inalámbrica WI-FI como la mas óptima, en la Tabla 3.8 se puede observar los módulos WI-FI existentes en el mercado con sus respectivos parámetros técnicos.

Tabla 3.8: Comparación de las características de los modulos Wifi.

PARÁMETROS	Arduino WiFi Shield	ESP-01	ESP8266-WM1
Estandar	802.11 b/g	802.11 b/g/n	802.11 b/g/n
Potencia de Tx	17 dBm	19.5 dBm	19.5dBm
Umbral de recepción	-88 dBm	-91 dBm	-93 dBm
Frecuencia de Reloj:	Según el microprocesador que se utilice	80MHz/ 160MHz	80MHz/ 160MHz
Memoria Flash Externa:	Tarjeta Micro SD	4MB	4MB
Consumo de corriente	250 mA	–	400mA
Pines Digitales GPIO:	12	4	11
Pines Analógico ADC:	Según el microprocesador que se utilice	–	1 (0 - 1V)
Costo	Alto	Bajo	Medio
Voltaje de Entradas/Salidas:	3.3Vdc, 5Vdc	3.3V DC (No usar 5V)	3.3V DC
Voltaje de Alimentación	5V	3.3V DC	5V DC

Elaborado por: El investigador, en base a [11], [32]

Los dispositivos mencionados comparten características similares, por lo cual se considero los parámetros de potencia de transmisión, costo y consumo de corriente como los factores mas importantes, la tarjeta Shield WI-FI de Arduino posee la menor potencia de transmisión además de un costo elevado, el módulo ESP8266-WM1 ofrece mejores beneficios en cuanto a cobertura y consumo de corriente y a un costo medio, sin embargo adquirir este modulo en el mercado dificulta al desarrollo del proyecto, por lo cual el modulo seleccionado es el ESP-01 que se muestra en la Figura 3.10 que tiene características similares al ESP8266-WM1, a diferencia del costo y la adquisición del mismo fácilmente en el mercado.

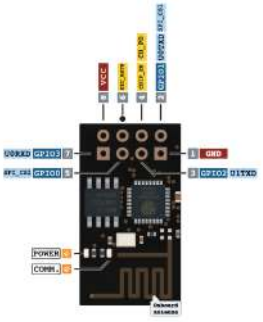


Figura 3.10: Modulo ESP-01 [11].

3.7 Construcción de prototipo del sistema de monitoreo apícola.

Para la construcción del prototipo apiario se basa en las colmenas movilistas que son aquellos que presentan unos cuadros móviles de madera, en el interior de la colmena, sobre los que se sitúan los panales, en ellos se coloca una capa de cera estampada (lámina de cera), las abejas construyen el panal, estirándola y añadiendo más cera, se conoce como cera estirada, dentro de la colmena movilistas existen infinidad de tipos los más frecuentes son: colmenas Layens, Langstroth y Dadant, la que se uso para la construcción del prototipo fue la colmena Langstroth que son colmenas de crecimiento vertical la cual ha sido adaptada y modificada para integrar los sensores de temperatura, humedad, dióxido de carbono, peso y presencia de abejas dentro de la colmena, en la Figura 3.11 se muestra las partes que conforman la colmena Langstroth.

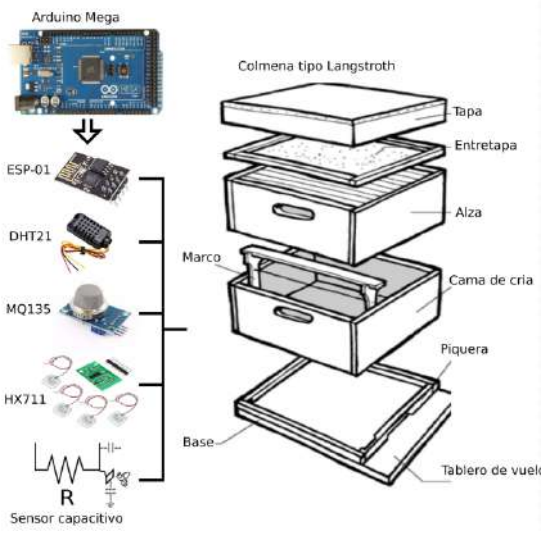
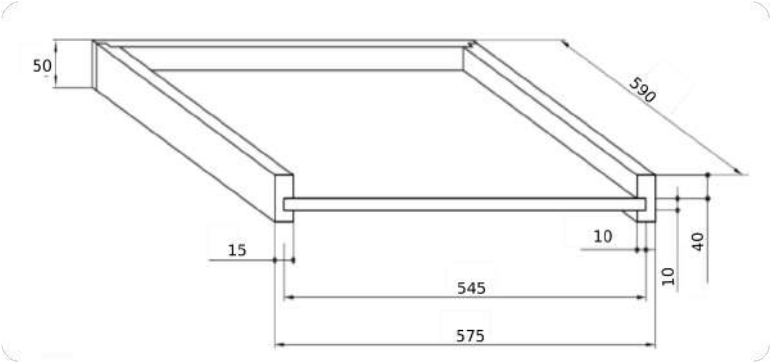


Figura 3.11: Colmena Langstroth
Elaborado por: El investigador, en base a [53]

Base: La base de la colmena se creó a partir de una tabla de madera rectangular de medidas de 575 mm x 590 mm, con un grosor de 15 mm, se le añade tres tiras de madera de unión tipo encastre y unidas entre ellas con unión a media madera. El lado delantero no lleva tira puesto que es la pista de aterrizaje de las abejas para entrar y salgan de la colmena, en la Figura 3.12 se muestra las medidas y el terminado de la base de la colmena.

(a) Base de la colmena

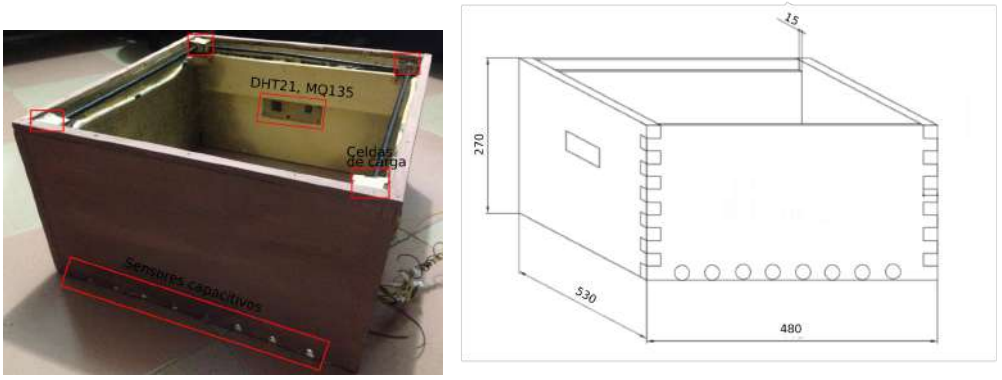


(b) Base de la colmena (medidas en mm)

Figura 3.12: Construcción de la base de la colmena.
Elaborado por: El investigador, en base a [53]

Cámara de cría: : La cámara de cría se creó con 2 tablas de madera rectangulares de valores de 480 mm x 270 mm x 15 mm y 2 tablas de 530 mm x 270 mm x 15 mm que se unirán con unión tipo machihembrado. Además es donde se colocan los sensores (Temperatura, Humedad, Co2, Peso, Población), para el cual se debe hacer pequeñas modificaciones en la cámara de cría para albergar dichos sensores, en las esquinas se colocan cuatro ranuras de 35 mm x 35 mm para colocar las celdas de carga (Peso de la colmena), en la parte frontal se realizaron ocho agujeros de 15 mm de

radio para albergar los sensores capacitivos (densidad de población) y finalmente en la parte posterior en el centro de la tabla (480 mm x 270 mm) se colocaron los sensores DHT21, MQ135 (Temperatura/Humedad, Co2), tal y como se muestra en la Figura 3.13a.



(a) Cámara de cría (b) Cámara de cría (medidas en mm)

Figura 3.13: Construcción de la cámara de cría.
Elaborado por: El investigador, en base a [53].

En la Figura 3.14 se muestra el diagrama de implementación del circuito electrónico y sus conexiones de los sensores al microcontrolador ATmega2560.

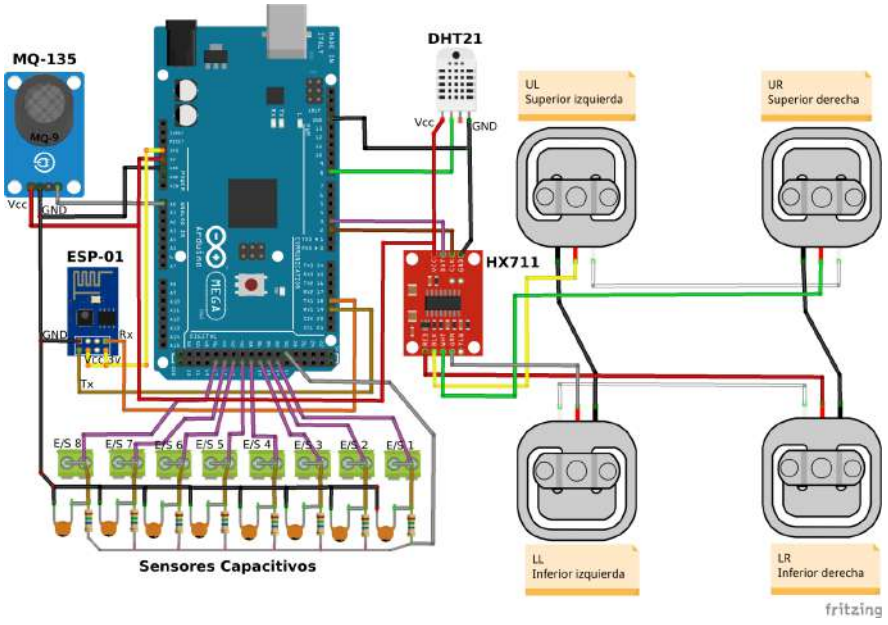


Figura 3.14: Diagrama de implementación del sistema de monitoreo apícola
Elaborado por: El investigador.

Las conexiones necesarias para el uso de los sensores se muestran en las Tablas 3.9 hasta la Tabla 3.15, además de cada uno de los pines a los que deben ir conectados.

Tabla 3.9: Conexión entre la celda de carga y el módulo HX711

Celda de carga	Módulo HX711
Cable rojo	Pin E+
Cable amarillo	Pin E-
Cable verde	Pin A-
Cable gris	Pin A+

Elaborado por: El investigador.

Tabla 3.10: Conexión entre el módulo HX711 y Arduino

Módulo HX711	Arduino MEGA
Pin GND	Pin GND
Pin DT	Pin 3
Pin SCK	Pin 2
Pin VCC	Pin 5 V

Elaborado por: El investigador.

Tabla 3.11: Conexión entre el sensor MQ135 y Arduino

Sensor MQ135	Arduino MEGA
Pin GND	Pin GND
Pin A0	Pin A0
Pin D0	———
Pin VCC	Pin 5 V

Elaborado por: El investigador.

Tabla 3.12: Conexión entre el sensor MQ135 y Arduino

Sensor DHT21	Arduino MEGA
Pin GND	Pin GND
Pin Data	Pin 6
Pin D0	———
Pin VCC	Pin 5 V

Elaborado por: El investigador.

Tabla 3.13: Conexión entre el modulo ESP01 y Arduino

Modulo ESP01	**Arduino MEGA**
Pin GND	Pin GND
Pin Tx	Pin Rx1
Pin Rx	Pin Tx1
Pin VCC	Pin 3V3

Elaborado por: El investigador.

Tabla 3.14: Conexión entre el modulo MAX6675 y Arduino

Modulo MAX6675	**Arduino MEGA**
Pin GND	Pin GND
Pin CS	Pin 6
Pin SO	Pin 5
Pin SCK	Pin 7
Pin VCC	Pin 5V

Elaborado por: El investigador.

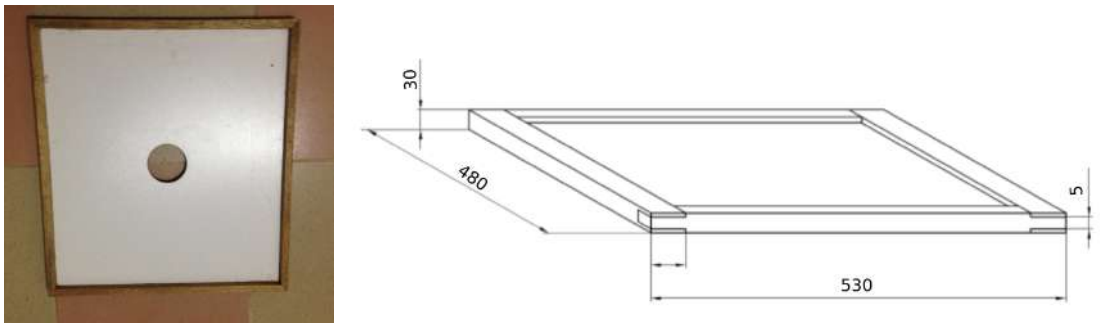
Tabla 3.15: Conexión entre el sensor capacitivo y Arduino

Sesor Capacitivo		**Arduino MEGA**
Pin de envio	E0	Pin 30
Pin de recepcion	E1	Pin 31
	E2	Pin 33
	E3	Pin 35
	E4	Pin 37
	E5	Pin 39
	E6	Pin 41
	E7	Pin 43
	E8	Pin 45
Pin de envio	S0	Pin 32
Pin de recepcion	S1	Pin 34
	S2	Pin 36
	S3	Pin 38
	S4	Pin 40
	S5	Pin 42
	S6	Pin 44
	S7	Pin 46
	S8	Pin 48

Elaborado por: El investigador.

Entretapa: La entretapa se creó de una tabla de madera enmarcada, la madera central está encastada al marco 10 mm y tiene las dimensiones siguientes: 530 mm

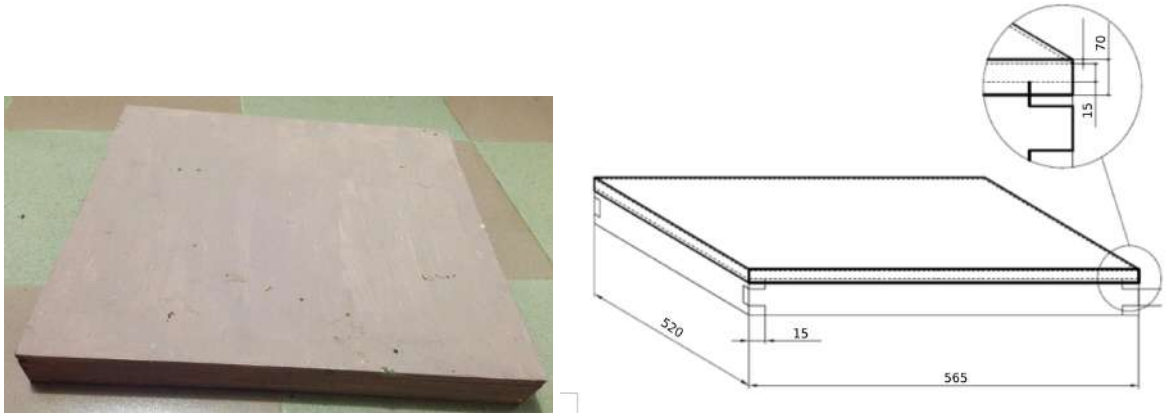
de largo, 480 de ancho y 15 mm de grosor, los marcos están unidos con unión tipo machihembrado las medidas y el acabado de la entretapa se aprecia en la Figura 3.15.



(a) Entretapa (b) Entretapa (medidas en mm)

Figura 3.15: Construcción de la entretapa de la colmena.
Elaborado por: El investigador, en base a [53].

Tapa: La tapa de la colmena se creo de una lámina de madera de 70 mm de espesor y de 565 mm x 520 mm, unida mediante junta plana a cuatro regletas como marco, dos para el lado largo y dos para el corto con las medidas indicadas en la Figura 3.16.

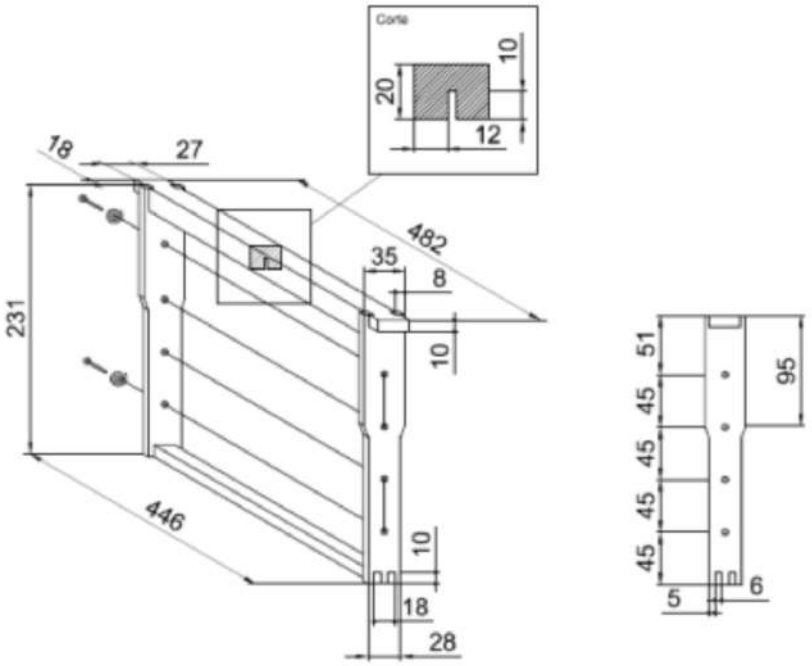


(a) Tapa de la colmena (b) Tapa de la colmena (medidas en mm)

Figura 3.16: Construcción de la tapa de la colmena.
Elaborado por: El investigador, en base a [53].

Marcos o cuadros: Los marcos se creo con tablas rectangulares como marco, pero cada lado con dimensiones distintas, como se puede observar en la Figura 3.17b todas las medidas son estándares en la construcción de marcos de colmenas, en total una cámara de cría consta de 10 marcos como mínimo.

(a) Marco o cuadro de madera



(b) Marco o cuadro de madera (medidas en mm)

Figura 3.17: Construcción de marco o cuadro de madera.
Elaborado por: El investigador, en base a [53]

El cambio del apiario por el prototipo construido por el investigador se debe seguir el manual básico de apicultura, considerando, el equipo de protección, la instalación del apiario y los pasos para la revisión de la colmena, como de medidas de seguridad, que se describen en el anexo 9.

3.7.1 Prueba de funcionamiento de los sensores.

Se realizaron diferentes pruebas y mediciones, para verificar el funcionamiento de los los sensores en el prototipo de monitoreo apícola. Para el funcionamiento de los modulo HX711, MQ135, DHT21 y MAX6675, se deben incluir la librería para cada

modulo respectivo en el software Arduino, el cual facilita la interpretación de los datos emitidos por los sensores.

En el modulo HX711 primero es necesario hallar el valor de la escala, es decir hallar el factor de conversión para convertir valor de lectura en un valor con unidades de peso, se consigue un objeto con peso conocido cercano al valor máximo de medición de la celda de carga, es decir 5kg, En la Fgura3.18 se muestra el factor de calibración para calcular la escala del sensor mediante la siguiente ecuación.

$$Escala = \frac{Valor\_promedio\_recibido}{Peso\_real}$$

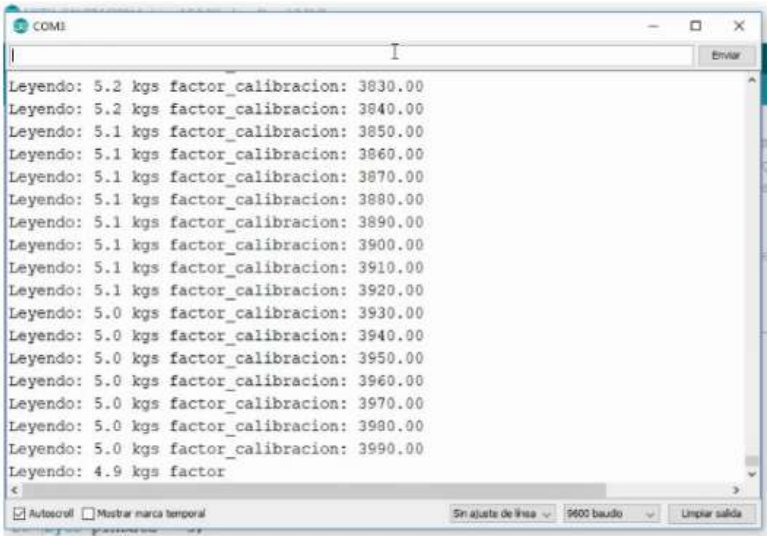


Figura 3.18: Factor de calibración del sensor de peso a través del monitor serial de Arduino

Elaborado por: El investigador.

$$Escala = \frac{3990}{5} = 798$$

Como resultado de esta calibración se obtuvo la escala para la medición del peso de la colmena máximo de 5kg

Para el sensor MQ135 el rango de detección va desde 10 a 1000ppm (partes por millón) y lo clasifica según el tipo de gas a detectar, para el sistema de monitorio es necesario medir la concentración de dióxido de carbono dentro de la colmena, y su medida según las especificaciones del sensor va des de los 74ppm hasta los 350ppm, para la detección de CO2.

La verificación de la lectura de los sensores se muestra en la Figura 3.19 en donde se puede observar que los niveles de temperatura, humedad, dióxido de carbono y peso de la colmena son aceptables.

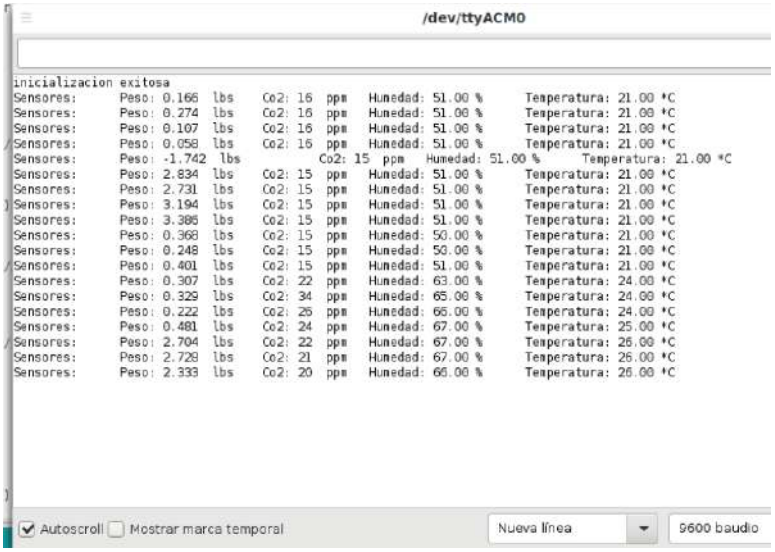


Figura 3.19: Prueba de funcionamiento de los sensores DHT21, HX711 y MQ135
Elaborado por: El investigador.

Para el sensor capacitivo se utilizo la librería CapacitiveSensor de arduino, el cual cambia un pin de envío del microcontrolador a un nuevo estado y luego espera que el pin de recepción cambie al mismo estado que el pin de envío. Una variable se incrementa dentro de un tiempo de bucle en cuanto cambia de estado el pin de recepción, luego el método informa el valor de la variable, que está en unidades arbitrarias en la Figura 3.20 se muestra las unidades arbitrarias del sensor.

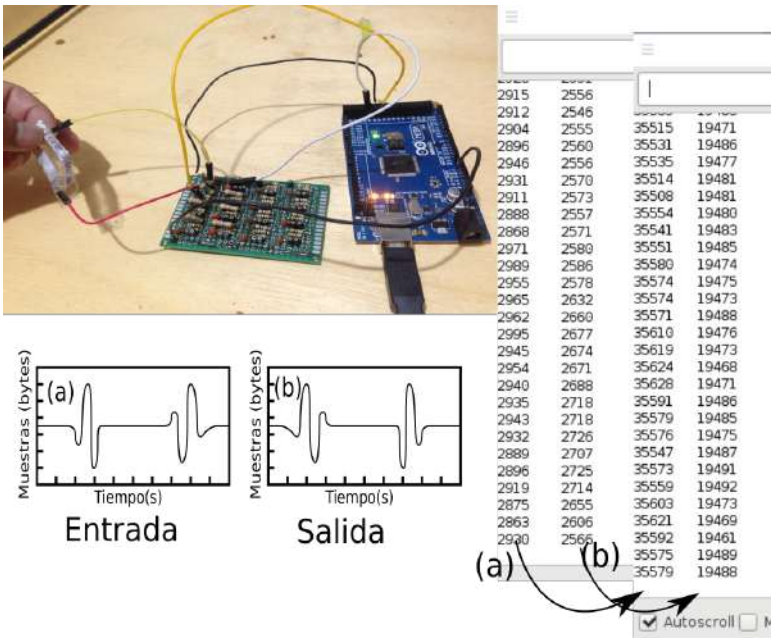


Figura 3.20: Prueba de funcionamiento del sensor capacitivo.
Elaborado por: El investigador

3.8 Planificación y requerimientos del diseño del software.

El diseño del sistema de monitoreo apícola se lo puede observar en la Figura 3.21, el cual consta de tres partes. La primera parte consta de la adquisición y procesamiento de los datos, a través de los sensores de temperatura, húmeda, dióxido de carbono, peso y capacitivos. La segunda parte es la conexión del sistema a la red wifi mas cercana y enviar los datos al servidor local donde el usuario puede acceder al sistema y ver la información. Por ultimo, la tercer parte es el servidor local donde se almacena toda la información enviada por los sensores a la base de datos con el fin de mostrar los datos en tiempo real y almacenarlos localmente, con los datos almacenados se extraen y se ejecuta en la interfaz de programación de aplicaciones (API) el cual consta de herramientas para el desarrollo del aprendizaje automático.

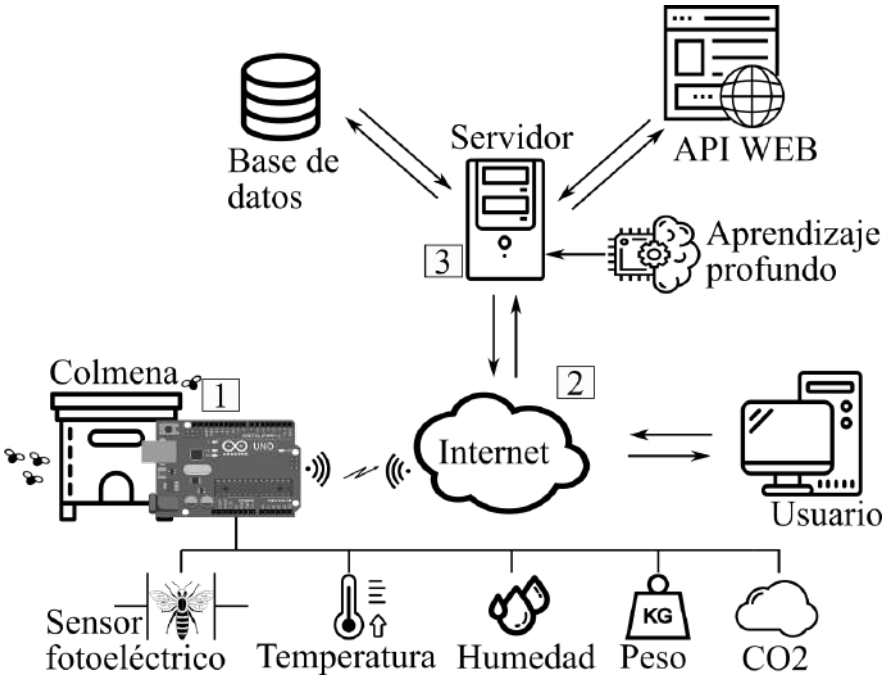


Figura 3.21: Diseño de implementación del sistema de monitoreo apícola.
Elaborado por: El investigador.

3.8.1 Selección del framework web para Python

Framework es un entorno de trabajo, es el esquema o estructura que se establece y que se aprovecha para desarrollar y organizar un software determinado, existen varios frameworks basados en Python para dar acceso a los usuarios información detallada contenida en base de datos, además de, crear una interfaz de programación de aplicaciones conocida también por sus siglas API, en inglés, *"application programming interface"* de manera rápida y simple, utilizando protocolos HTTP para la comunicación, en la Tabla 3.16 se puede muestra una comparación entre frameworks basados en lenguaje de programación Python para el desarrollo de la aplicación del sistema de monitoreo apícola.

Tabla 3.16: Comparación de los servidores web basados en Python.

Parámetros	Django	Flask	Web2py
Descripción	Django es un framework web de Python de alto nivel que fomenta el desarrollo rápido y el diseño limpio y pragmático	Flask es un framework ligero de aplicación web	Web2py es un framework de código libre para el desarrollo rápido de aplicaciones web basadas en bases de datos rápidas, escalables, seguras y portátiles.
Ventajas	• El desarrollo de un prototipo simple puede ser muy rápido. • Documentación de primer nivel y ayuda de la comunidad. • Organización MVC clara y definida. • Administración simple de bases de datos. • Altamente personalizable	• Minimalista sin perder poder. • Muchos recursos disponibles en línea. • Extremadamente fácil de construir un prototipo rápido. • Muy flexible. • Gran documentación	• La documentación está escrita en forma de libro que es bueno para principiantes. • Incluye un IDE basado en la web para crear y administrar aplicaciones. • Se puede ajustar para ser realmente rápido en producción. • Fácil de aprender sin perder potencia.
Desventajas	• El enrutamiento requiere cierto conocimiento de las expresiones regulares. • La documentación no cubre escenarios del mundo real. • Los errores de plantilla fallan silenciosamente por defecto.	• No está explícitamente diseñado para manejar la programación asíncrona. • La creación de un proyecto grande requiere un conocimiento previo del marco. • Orientado a HTML, no orientado a API	• El IDE web no es un IDE con todas las funciones.

Elaborado por: El investigador, en base a [54].

El framework seleccionado para la creación de la aplicación web es Flask, debido a sus ventajas puesto que es flexible, ligero, y minimalista para la creación de APIs, además de que su documentación es amplia y fácil de entender para quienes empiezan a desarrollar en Python desde cero, en comparación a Django el cual se utiliza para proyectos más grandes y robustos, su documentación es muy extensa y tiende a ser confusa. Web2py pese a que es facil de entender y su documentación es buena, su comunidad no es muy activa en cuanto al desarrollo de aplicaciones, a demás de que posee fallas de conexión a bases de datos lo que dificulta al usuario crear aplicaciones.

Características de hardware del servidor:

El aprendizaje automático y el aprendizaje profundo en cualquier conjunto de datos de software se requiere un sistema informático lo suficientemente potente como para manejar grandes cantidades de información. por lo tanto las características de hardware que se utilizo para el desarrollo del sistema de monitoreo apícola son:

- Unidad central de procesamiento (CPU) : procesador Intel Core i7 3.40 GHz
- Memoria RAM : 8 GB.
- Unidad de procesamiento de gráficos (GPU) : NVIDIA GeForce GTX 960.
- Sistema operativo : Debian 10

3.8.2 Selección de la base de datos.

Existen varios gestor de bases de datos de código abierto, disponible para distintos sistemas operativos, como Linux, Mac OS X, Solaris, Windows y otros más, destinado al desarrollo web y para el almacenamiento de datos, en la Tabla 3.17 se analizan las características mas importantes de los tres gestores de datos mas relevantes usadas en el desarrollo de aplicaciones web.

Tabla 3.17: Comparación de las características de las bases de datos.

Parámetros	Sql Server	Postgres	MySql
Descripción	Es un sistema de gestión de base de datos relacional, desarrollado por Microsoft, su principal lenguaje de consulta es Transact-SQL	Es un gestor de bases de datos relacional orientado a objetos y de código abierto,	Es un sistema de gestión de base de datos relacional de código abierto, basado en lenguaje de consulta estructurado SQL.
Características	• Puede replicar todo tipo de datos. • Admite diferentes idiomas como: C, C, PHP, Python, etc. • No distingue entre mayúsculas y minúsculas, • Ofrecer el mecanismo para incorporarse con aplicaciones web . • Compatibilidad con aplicaciones desarrollada en .Net.	• Es compatible con el control de concurrencia de versiones múltiples (MVCC). • Distingue entre mayúsculas y minúsculas. • Admite imágenes, videos, almacenamiento de audio y también admite datos gráficos. • Se ejecuta en todos los sistemas operativos.	• Totalmente administrable y escalable con alta disponibilidad y seguridad sin costo adicional • Seguras y están bloqueadas con contraseñas encriptadas. • Se ejecuta en segundo plano y gestiona solicitudes de clientes MySQL. • MySQL se basa en un modelo cliente-servidor.

Elaborado por: El investigador, en base a [55].

La base de datos que se tomó en consideración para el desarrollo de la aplicación fue Mysql, debido a su gran capacidad de manejar sin problemas millones de registros de datos ideal para el desarrollo del aprendizaje profundo en donde se requiere grandes cantidades de información para el entrenamiento de redes neuronales.

3.8.3 Instalación de librerías y herramientas para el desarrollo del aprendizaje automático.

Instalación del framework web Flask

Entorno virtual: Los entornos virtuales son grupos independientes de bibliotecas de

Python para administrar las dependencias de su proyecto, de tal manera que el proyecto no afecte a otros proyectos ni a los paquetes del sistema operativo. Para crear un entorno virtual se siguen los siguientes comandos en la terminal del sistema operativo Debian.

```
Python + pip.
# Debian, Ubuntu:
$ sudo apt-get install python3-pip
Luego instale virtualenv
$ sudo apt-get install python3-virtualenv
Crear un entorno virtual:
$ virtualenv -p python3 env
Active su entorno virtual:
$ source env/bin/activate
(env) $ ....
Framework Flask:
(env) $  pip install Flask
(env) $ mkdir BeeApp.py
```

Figura 3.22: Archivos para crear la aplicación web
Elaborado por: El investigador

En la Figura 3.22 se puede observar los archivos necesarios para crear la aplicación web del sistema de monitoreo apícola, en el cual se describe a continuación:

- env: es el entorno virtual creado para contener los paquetes instalados para ejecutar la aplicación.
- static: contiene el estilo y los módulos del diseño HTML,
- templates: guarda las plantillas del diseño HTML en el navegador del usuario.
- BeeApp.py es la aplicación principal contiene las rutas de activación de funciones escritas en lenguaje de programación Python.
- BaseDatos.csv contiene los datos de los sensores registrados en la base de datos Mysql en formato .csv (del inglés comma-separated values, valores separados por comas) .
- requerimientos.txt contiene la lista de librerías y paquetes necesarios para compilar la aplicación principal.

Instalación de las librerías para el desarrollo del aprendizaje automático.

Jupyter Notebook, es un entorno de trabajo interactivo que permite desarrollar código en Python, utilizado ampliamente en análisis numérico, estadística, machine learning, entre otros.

```
(env) $  pip install jupyterlab
(env) $ jupyter notebook --allow-root
```

Al ejecutar Jupyter Notebook se imprime en la terminal un registro de actividades de ejecución de la interfaz, mostrando una dirección IP junto al puerto para acceder a la interfaz web de JupyterLab como se muestra en la Figura 3.23 el interprete de Jupyter Notebook se encuentra activo.

```
Output
[I 13:41:02.269 NotebookApp] The Jupyter Notebook is running at:
[I 13:41:02.269 NotebookApp] http://localhost:8888/?token=01de7785
bcc3fb813bdfe346f34b0a904cc5784fe81a85b1
[I 13:41:02.269 NotebookApp]  or http://127.0.0.1:8888/?token=01de778
5bcc3fb813bdfe346f34b0a904cc5784fe81a85b1
[I 13:41:02.269 NotebookApp] Use Control-C to stop this server and shut
down all kernels (twice to skip confirmation).
```

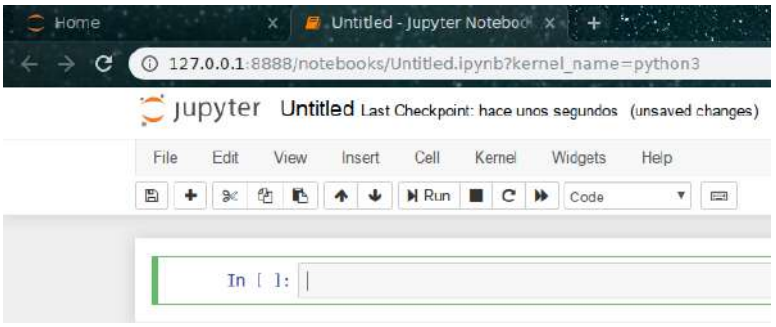


Figura 3.23: Interfaz de entorno de trabajo Jupyter Notebook.
Elaborado por: El investigador

En la interfaz web de Jupyter Notebook se instala las librerías y bibliotecas necesarias para el aprendizaje automático, bajo el sistema de gestión de paquetes pip.

```
$ pip install tensorflow  keras mxnet theano cntk lasagne pandas
matplotlib numpy
```

Cada librería tiene sus propios beneficios y limitaciones para el desarrollo del aprendizaje automático, además se consideran como las mejores para desarrollar modelos sofisticados y motores de predicción.

Prerrequisitos para la instalación de la base de datos MySql

Apache, mysql, php y phpmyadmin, está disponible dentro de los repositorios de software predeterminados del sistema operativo Debian y sus derivados, lo que permite instalarlo utilizando herramientas convencionales de administración de paquetes ejecutando en la terminal los comandos:

```
$ sudo apt-get -y install apache2
$ sudo apt-get install phpmyadmin
Reinicie el servidor web apache.
$ sudo systemctl restart apache2
Para acceder a phpMyAdmin desde un navegador visite:
http://localhost/phpmyadmin/index.php
$ sudo apt -y install wget php php-cgi php-mysqli php-pear php-mbstring
php-gettext libapache2-mod-php php-common php-phpseclib php-mysql
$ sudo apt install mariadb-server mariadb-client
$ sudo systemctl status mariadb # verifica si el servicio mariadb se
encuentra habilitado.
```

En la Figura 3.24 se muestra el servicio de mariadb activo y se esta ejecutando para iniciarse automáticamente en el arranque del sistema.

```
● mariadb.service - MariaDB 10.3.22 database server
   Loaded: loaded (/lib/systemd/system/mariadb.service; enabled; vendor preset: enabled)
   Active: active (running) since Mon 2020-05-11 23:58:43 -05; 7s ago
     Docs: man:mysqld(8)
           https://mariadb.com/kb/en/library/systemd/
  Process: 10619 ExecStartPre=/usr/bin/install -m 755 -o mysql -g root -d /var/run/mysql
  Process: 10620 ExecStartPre=/bin/sh -c systemctl unset-environment _WSREP_START_POSITI
  Process: 10622 ExecStartPre=/bin/sh -c [ ! -e /usr/bin/galera_recovery ] && VAR= ||
  Process: 10703 ExecStartPost=/bin/sh -c systemctl unset-environment _WSREP_START_POSIT
  Process: 10705 ExecStartPost=/etc/mysql/debian-start (code=exited, status=0/SUCCESS)
 Main PID: 10672 (mysqld)
   Status: "Taking your SQL requests now..."
    Tasks: 31 (limit: 4915)
   Memory: 71.7M
   CGroup: /system.slice/mariadb.service
           └─10672 /usr/sbin/mysqld
```

Figura 3.24: Verificación del servicio de la base de datos MySql.
Elaborado por: El investigador

Se puede comprobar el funcionamiento de PhpMyadmin, colocando en el navegador la dirección IP del servidor local, seguido de PhpMyadmin, quedando http://192.168.1.5/phpmyadmin/, lo cual visualizó la página principal de PhpMyadmin, como se observa en la Figura 3.25.

Figura 3.25: Página de ingreso de PhpMyadmin.
Elaborado por: El investigador

Creación de la base de datos local

Se crear una base de datos local con el nombre “BeeApp” para almacenar los datos dentro de la tabla “datossensores” que registran los datos emitidos por los sensores del sistema de monitoreo apícola, dentro de la tabla de la base de datos se agregaron 7 variables que son:

- Id
- Fecha
- Temperatura
- Humedad
- CO2
- Peso
- Población

En donde el “Id” contiene la identidad de cada dato que se agrega a la tabla, la variable “Fecha” se agregó para conocer en qué momento se añadió el dato y por último las cinco variables a medir que son: temperatura, humedad, dióxido de carbono, peso y la población de la colmena, cada una de las variables a medir se declararon tipo flotante y con argumento (5,2), es decir que pueden tomar un valor de 5 dígitos, en donde tres dígitos son enteros y dos son decimales, la variable Fecha se asignó la herramienta CURRENT_TIMESTAMP la cual entrega la fecha y hora del momento en que se ingresa un dato. La estructura de la base de datos del servidor local creada se puede observar en la Figura 3.26 con mayor detalle.

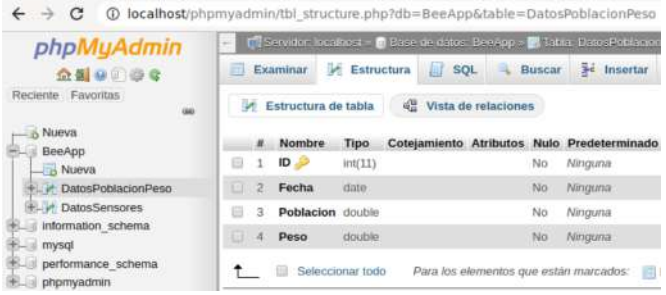


Figura 3.26: Estructura de la base de datos del servidor local.
Elaborado por: El investigador

Posteriormente, se realizó la conexión entre el modulo ESP-01 y el servidor local para que se almacene la información enviada por los sensores, para ello se utilizo el lenguaje de programación PHP para la conexión con la base de datos, se creó un archivo denominado conexion.php como se muestra en la siguiente Figura 3.27, en donde se asigna el host, el nombre de la base, el usuario y la contraseña para acceder a la base de datos mysql.

```php
<?php
class conexion
{
        private $servidor;
        private $usuario;
        private $contrasena;
        private $basedatos;
        public  $conexion;

        public function __construct(){
                $this->servidor   = "localhost";
                $this->usuario    = "phpmyadmin";
                $this->contrasena = "debian";
                $this->basedatos  = "phpmyadmin";

        }

        function conectar(){
                $this->conexion = new PDO("mysql:host=$this->servidor;
                                           dbname=$this->basedatos",
                                           "$this->usuario",
                                           "$this->contrasena");
        }

        function cerrar(){
                $this->conexion->close();
        }
}
?>
```

Figura 3.27: Archivo para abrir conexión con la base de datos local.
Elaborado por: El investigador

Luego de iniciar la conexión al servidor, el modulo ESP-01 realiza una consulta SQL utilizando el método HTTP GET , se creó el archivo enviar_datos.php como se muestra en la Figura 3.28 para inserta los valores de los sensores de temperatura, humedad, dioxido de carbono, peso y presencia en la tabla “datossensores” , con el fin de tomar muestras de datos para el entrenamiento de la red neuronal artificial.

```php
<?php
class Herramienta{
        private $conexion;

        function __construct(){
                require_once("conexion.php");
                $this->conexion = new conexion();
                $this->conexion->conectar();
        }

        public function ingresar_datos($temperatura_php, $humedad_php, $co2_php, $peso_php, $poblacion_php){
                $sql = " insert into datossensores values (null, now(), ?, ?, ?, ?, ?) ";
                $stmt = $this->conexion->conexion->prepare($sql);

                $stmt->bindValue(1, $temperatura_php);
                $stmt->bindValue(2, $humedad_php);
                $stmt->bindValue(3, $co2_php);
                $stmt->bindValue(4, $peso_php);
                $stmt->bindValue(5, $poblacion_php);

                if($stmt->execute()){
                        echo "Ingreso Exitoso";
                }else{
                        echo "no se pudo registrar datos";
                }
        }
}
?>
```

Figura 3.28: Archivo para ingresar la información a la base de datos.
Elaborado por: El investigador

El modulo ESP-01 enviará la información de los sensores a la base de datos en el servidor local, con el fin de tomar muestras de los sensores del sistema de monitoreo apícola en periodos cortos de tiempo, en la Tabla 3.18 se observan los datos enviados al servidor local, cabe recalcar que los datos son de prueba los valores reales se indican en la Figura 3.31 y en la Figura 3.32.

Tabla 3.18: Datos enviados al servido local

Fecha	Temperatura	Humedad	CO2	Peso	Poblacion
2019-12-01 07:00:00	35.2	66	145	2.46	37
2019-12-01 07:05:00	34.5	67	146	2.46	78
2019-12-01 07:10:00	34.7	67	147	2.46	156
2019-12-01 07:15:00	34.1	66	147	2.46	321
2019-12-01 07:20:00	33.4	64	145	2.44	642
2019-12-01 07:25:00	33.7	64	145	2.44	1289
2019-12-01 06:30:00	33.3	63	143	2.44	2578
2019-12-01 07:35:00	33.9	63	142	2.44	5164
2019-12-01 07:40:00	34.5	63	142	2.45	10331
2019-12-01 07:45:00	34.8	64	140	2.45	14598
2019-12-01 07:50:00	34.2	64	140	2.45	17851
2019-12-01 07:55:00	34.9	63	140	2.45	21
2019-12-01 08:00:00	35.7	63	145	2.45	57
2019-12-01 08:05:00	35.5	62	145	2.45	236
2019-12-01 08:10:00	36.7	62	142	2.46	472
2019-12-01 08:15:00	36.3	63	142	2.46	1067
2019-12-01 08:20:00	37.4	63	142	2.46	2134
2019-12-01 08:25:00	37.6	63	142	2.47	4268
2019-12-01 08:30:00	37.8	64	147	2.47	8657
2019-12-01 08:35:00	37.9	65	147	2.47	123
2019-12-01 08:40:00	36.7	65	148	2.47	596
2019-12-01 08:45:00	34.5	65	148	2.47	1542
2019-12-01 08:50:00	34.7	66	153	2.47	3434
2019-12-01 08:55:00	34.1	66	153	2.47	7218
2019-12-01 09:00:00	33.4	67	152	2.48	14786
2019-12-01 09:05:00	33.3	66	152	2.48	53
2019-12-01 09:10:00	33.9	64	152	2.48	117
2019-12-01 09:15:00	34.5	64	155	2.48	584
2019-12-01 09:20:00	34.8	63	155	2.48	1518
2019-12-01 09:25:00	34.2	63	155	2.44	3386
2019-12-01 09:30:00	34.9	63	155	2.44	6772
2019-12-01 09:35:00	35.7	64	153	2.45	13894
2019-12-01 09:40:00	35.1	64	153	2.45	14223
2019-12-01 09:45:00	35.7	63	152	2.45	14352
2019-12-01 09:50:00	35.2	63	152	2.46	14711
2019-12-01 09:55:00	35.5	62	152	2.46	15386
2019-12-01 10:00:00	35.7	62	156	2.46	16325
2019-12-01 11:05:00	35.1	63	156	2.46	138
2019-12-01 10:10:00	35.5	63	156	2.5	626
2019-12-01 10:15:00	36.7	63	155	2.5	1375
2019-12-01 10:20:00	36.3	64	153	2.55	3139
2019-12-01 10:25:00	37.4	65	153	2.56	6628

Elaborado por: El investigador.

3.9 Redes neuronales para la predicción de series temporales.

Las aplicaciones que utilizan redes neuronales artificiales para la predicción de series temporales, son expuestos por diversos autores que describen algoritmos basados en el aprendizaje automático como una herramienta para solucionar diversos problemas, por lo cual se recopilo los algoritmos mas relevantes según el tipo de aprendizaje y su respectivas variantes, como se muestra en la Figura 3.29, para posteriormente seleccionar el mas adecuado a la solución de la contextualización del problema propuesto.

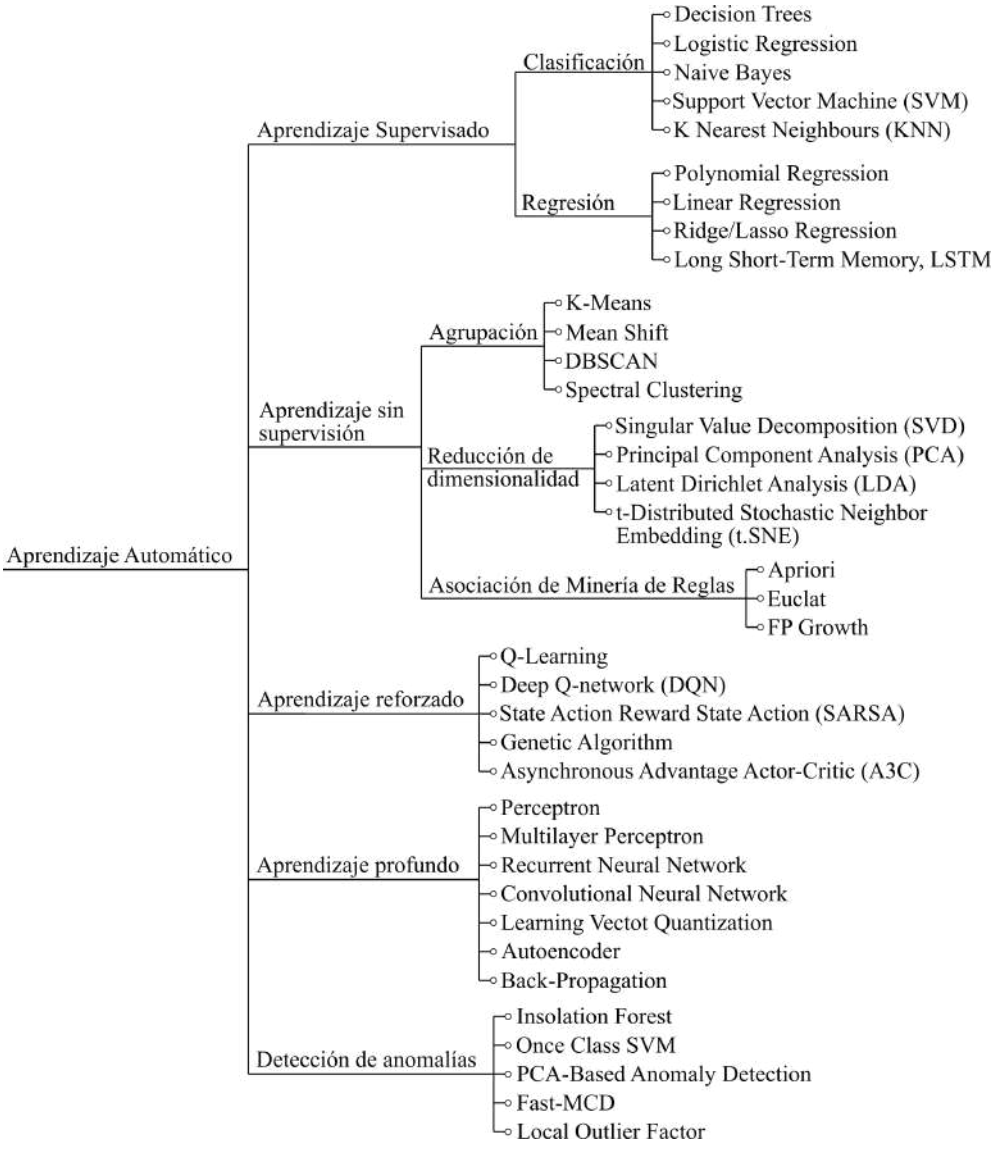


Figura 3.29: Algoritmos en el aprendizaje automático.
Elaborado por: El investigador, en base a [56]

Los algoritmos del aprendizaje automático, comprenden sus propias limitaciones y configuraciones para dar solución al problema, en ocasiones el algoritmo seleccionado según el tipo de aprendizaje no siempre es el adecuado, aveces resulta ser una combinación entre el tipo de aprendizaje y sus variantes con el fin de crear un modelo que cumplan las necesidades del problema, por el cual se creo un modelo de red neuronal con el objetivo de predecir la variación de población apícola y la predicción futura de las variables de temperatura, humedad, dióxido de carbono y peso de la colmena. En el anexo 10 se puede apreciar los diferentes tipos de modelos de redes neuronales.

3.9.1 Selección del modelo neuronal.

El sistema propuesto requiere realizar predicciones futuras utilizando los métodos de aprendizaje automático, como es el aprendizaje profundo, el tipo de problema del modelo esta relacionado con la predicción de series de tiempo, denominados redes neuronales recurrentes, en especifico el modelo de memoria a corto y largo plazo, o LSTM en el cual se muestra en la Figura3.30 el cual es un tipo de red neuronal recurrente utilizada en el campo del aprendizaje profundo, capaz de modelar sin problemas múltiples variables de entrada, como un conjunto de datos de series de tiempo, los valores futuros son dependientes de los valores pasados para predecir su comportamiento.

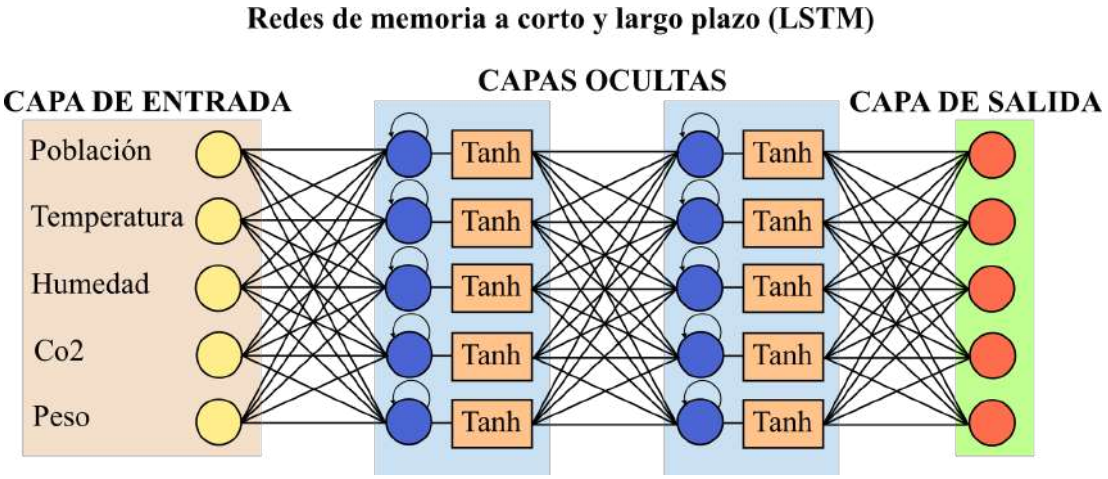


Figura 3.30: Memoria a corto y largo plazo, LSTM
Elaborado por: El investigador, en base a [57]

3.9.2 Red LSTM para la regresión.

Una vez definido la estructura del modelo de la red, se implementa el modelo seleccionado en el prototipo, para posteriormente realizar el entrenamiento y validación del desempeño en la predicción, se utilizó un conjunto de datos correspondientes a los días de Diciembre de 2019 hasta Abril del 2020, con un total de 152 muestras para el entrenamiento y validación de la red neuronal, a continuación se detalla como preparar el conjunto de datos y la estructura del modelo en Python.

El vector de entrada hacia la red esta formado por el siguiente conjunto de datos: fecha, hora, temperatura, humedad relativa, dióxido de carbono, peso y la densidad poblacional de la colmena, utilizando estos datos se enmarca el problema de pronóstico en el que, dadas las condiciones en la que se encuentra la colmena de los días/horas anteriores, se predicen las condiciones a los días/horas siguientes.

En la Figura 3.31 se muestran las primeras y ultimas cinco filas del conjunto de datos de: Temperatura, Humedad y CO2, de cada hora correspondientes a los días de Diciembre de 2019 hasta Abril de 2020, con un total de 3648 muestras.

```
# Librerias
from pandas import read_csv
from matplotlib import pyplot
df = read_csv('BaseDeDatos.csv') # carga el conjunto de datos
values = df.values
display(df.head(5)) #imprime las primeras 5 filas
display(df.tail(5)) #imprime las ultimas 5 filas
```

	Fecha	Temperatura	Humedad	CO2
0	2019-12-01 00:00:00	34	65	120
1	2019-12-01 01:00:00	34	65	122
2	2019-12-01 02:00:00	34	65	122
3	2019-12-01 03:00:00	34	65	124
4	2019-12-01 04:00:00	35	65	124

	Fecha	Temperatura	Humedad	CO2
3643	2020-04-30 19:00:00	32	65	143
3644	2020-04-30 20:00:00	32	64	140
3645	2020-04-30 21:00:00	33	64	138
3646	2020-04-30 22:00:00	33	64	135
3647	2020-04-30 23:00:00	33	65	135

Figura 3.31: Visualización de los datos de temperatura, humedad y CO2.
Elaborado por: El investigador

Los datos de la densidad de población y el peso de la colmena se muestra en la Figura 3.32 y son monitorizados cada día, para un total de 152 muestras correspondientes a los días de Diciembre de 2019 hasta Abril de 2020.

	Fecha	Población	Peso
0	2019-12-01	14553	2.46
1	2019-12-02	14567	2.46
2	2019-12-03	14551	2.46
3	2019-12-04	14595	2.46
4	2019-12-05	14556	2.46

	Fecha	Población	Peso
147	2020-04-26	15786	2.44
148	2020-04-27	15723	2.45
149	2020-04-28	15637	2.45
150	2020-04-29	15793	2.46
151	2020-04-30	15961	2.47

Figura 3.32: Visualización de los datos de la densidad de población y peso de la colmena.
Elaborado por: El investigador

Con los datos del formulario, se puede crear gráficas de cada serie y ver su fluctuación en el tempo.

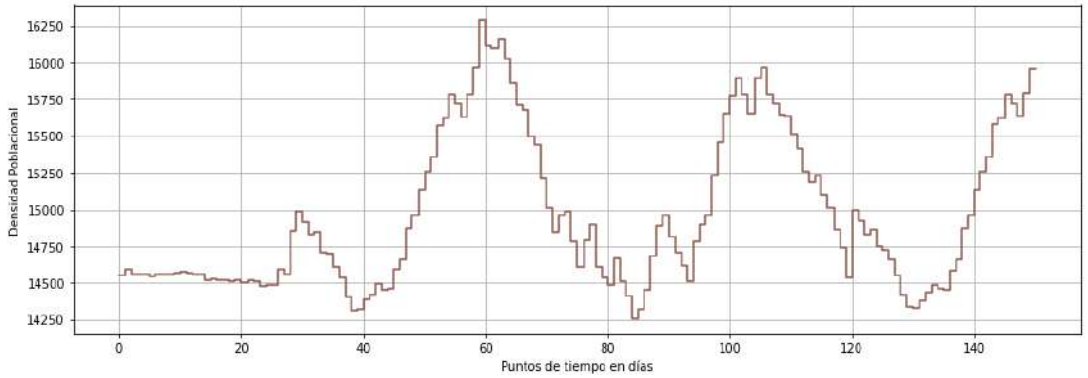


Figura 3.33: Fluctuación de la densidad poblacional de la colmena para el entrenamiento de la red neuronal artificial.

Elaborado por: El investigador.

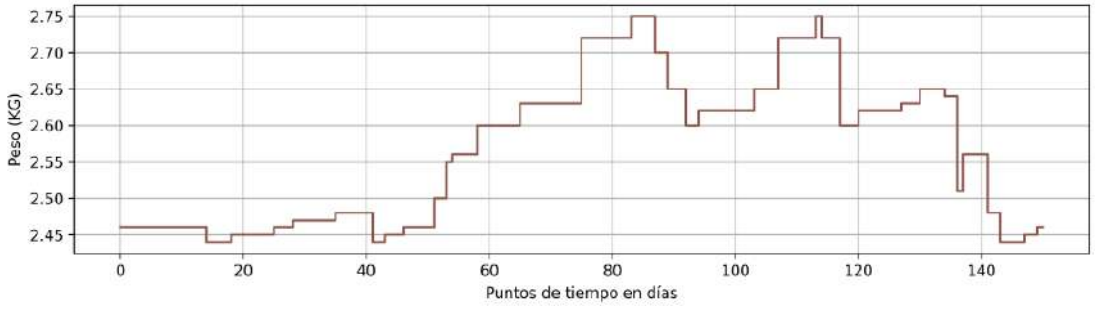


Figura 3.34: Fluctuación del peso de la colmena para el entrenamiento de la red neuronal artificial.

Elaborado por: El investigador.

Como se puede observar en las Figuras 3.33 y Figuras 3.34 los datos se distribuyen aproximadamente de manera normal, lo que significa que en un enfoque probabilístico basado en univariado sería un candidato decente para nuestro algoritmo de predicción.

La fluctuación de los datos registrados por los sensores de temperatura, humedad y Co2, son desmesurados como para poder observar a simple vista, debido al numero de datos correspondientes a 152 días cada hora, con un total de 3648 datos para entrenar el modelo de red, por lo cual se agruparan los datos en periodos de tiempo cortos, de esta manera poder percibir la fluctuación de los datos en el tiempo.

3.9.3 Preprocesado de datos.

En esta sección, se prepara el conjunto de datos de entrada para la red neuronal, para lo cual se enmarca los datos como un problema de aprendizaje supervisado como se muestra en la Figura 3.35, se normaliza las variables de entrada y lo convierte en un problema del tipo supervisado para alimentar a la red neuronal y poder entrenarla.

```
# conversión de series en aprendizaje supervisado
def series_to_supervised(data, n_in=1, n_out=1,␣
↪dropnan=True):
    """
    data: secuencia de observaciones como una lista o␣
↪matriz.
    n_in: Número de observaciones de retraso como␣
↪entrada (X).
    n_out: Número de observaciones como salida (y).
    dropnan: booleano si se deben soltar o no filas con␣
↪valores NaN.
    """
    n_vars = 1 if type(data) is list else data.shape[1]
    df = pd.DataFrame(data)
    cols, names = list(), list()
    # secuencia de entrada (t-n, ... t-1)
    for i in range(n_in, 0, -1):
        cols.append(df.shift(i))
        names += [('var%d(t-%d)' % (j+1, i)) for j in␣
↪range(n_vars)]
    # secuencia de pronóstico (t, t+1, ... t+n)
    for i in range(0, n_out):
        cols.append(df.shift(-i))
        if i == 0:
            names += [('var%d(t)' % (j+1)) for j in␣
↪range(n_vars)]
        else:
            names += [('var%d(t+%d)' % (j+1, i)) for j␣
↪in range(n_vars)]

    agg = pd.concat(cols, axis=1)
    agg.columns = names
    # soltar filas con valores NaN
    if dropnan:
        agg.dropna(inplace=True)
    return agg
```

Figura 3.35: Conversión de series en aprendizaje supervisado.
Elaborado por: El investigador.

La función MinMaxScaler transforma el rango de nuestros valores entre -1 y 1 como se muestra en la Figura 3.36 para normalizar y favorecer a nuestra red neuronal para

realizar los cálculos con precisión.

```
PASOS=7#Días para el entrenamiento
# Pre-cargar el conjunto de datos
values = df['Poblacion'].values
# transforma los datos en flotantes
values = values.astype('float32')
# normaliza la función
scaler = MinMaxScaler(feature_range=(-1, 1))
values=values.reshape(-1, 1)
scaled = scaler.fit_transform(values)
df['scaled'] = scaled
scaledMerge=df.drop('Poblacion',axis=1)
# se ajusta como aprendizaje supervisado
reframed = series_to_supervised(scaled, PASOS, 1)
reframed.head()
```

Figura 3.36: Función MinMaxScaler para normalizar los datos de entrada de la red.
Elaborado por: El investigador.

	var1(t-7)	var1(t-6)	var1(t-5)	var1(t-4)	var1(t-3)	var1(t-2)	var1(t-1)	var1(t)
7	-0.707821	-0.694048	-0.709788	-0.666502	-0.704869	-0.698967	-0.701918	-0.719625
8	-0.694048	-0.709788	-0.666502	-0.704869	-0.698967	-0.701918	-0.719625	-0.704869
9	-0.709788	-0.666502	-0.704869	-0.698967	-0.701918	-0.719625	-0.704869	-0.697983
10	-0.666502	-0.704869	-0.698967	-0.701918	-0.719625	-0.704869	-0.697983	-0.701918
11	-0.704869	-0.698967	-0.701918	-0.719625	-0.704869	-0.697983	-0.701918	-0.693064

Figura 3.37: Datos de entrada var1(t-7) a (t-1) y salida var1(t)
Elaborado por: El investigador.

Como se puede apreciar en la Figura 3.37 los datos de la densidad de población de la colmena, se encuentran normalizados y escalados listo para alimentar al modelo de red, de la misma manera se normalizan y se escalan los datos de temperatura, humedad, co2 y peso de la colmena, cambiando el conjunto de datos de entrada: values = df['Poblacion'].values por values = df['Temperatura'].values, values = df['Humedad'].values, values = df['Co2'].values y values = df['Peso'].values respectivamente.

3.9.4 Conjunto de datos de entrenamiento y validación.

Un modelo de red neuronal se construye como un modelo de regresión simple que recibe una entrada y suelta una salida, es decir, toma el valor de la densidad de población, temperatura, humedad, co2 y peso de la colmena de días anterior y predice los datos del día siguiente.

Para crear la red neuronal se divide los datos de la densidad poblacional de la colmena para entrenar *("train")* y validar *("test")*, utilizando una submuestra de 122 días (correspondientes al mes de Diciembre, Enero, Febrero y Marzo) par el entrenamiento de la red y una etapa de validación de los datos para la proyección de del mes Abril, correspondientes a un 80 % para el entrenamiento y 20 % para la validación de la muestra total de los datos en la Figura 3.38 se muestra el código para separa los datos para el entrenamiento y validación, en la Figura 3.39 se aprecia la fluctuación de la de la densidad poblacional de la colmena normalizados para el entrenamiento de la red neuronal.

```
#Se divide los datos de Entrenamiento y Validación
newReframed=reframed.drop(['var1(t)','var2(t)'],axis=1)
values = newReframed.values
n_train_days = 120
train = values[:n_train_days, :]
test = values[n_train_days:, :]
# entradas y salidas
x_train, y_train = train[:, :-1], train[:, -1]
x_val, y_val = test[:, :-1], test[:, -1]
# remodela la entrada [muestras, pasos de tiempo,␣
 ↪características]
x_train = x_train.reshape((x_train.shape[0], 1, x_train.
 ↪shape[1]))
x_val = x_val.reshape((x_val.shape[0], 1, x_val.
 ↪shape[1]))
print(x_train.shape, y_train.shape, x_val.shape, y_val.
 ↪shape)
```

Figura 3.38: División de datos para el entrenamiento y validación de la red.
Elaborado por: El investigador.

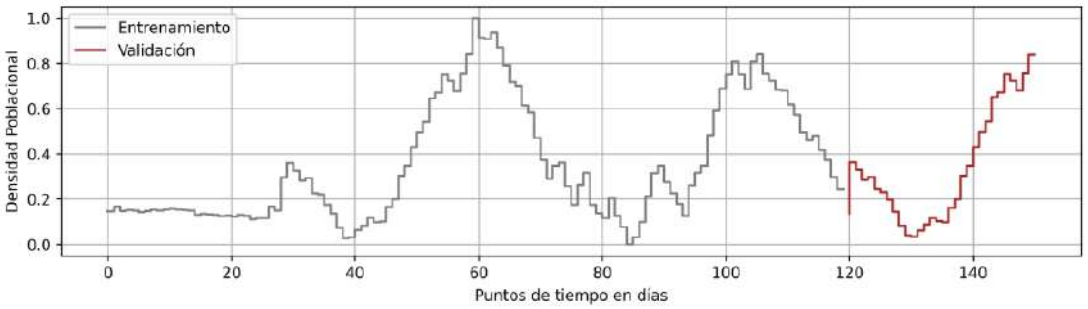


Figura 3.39: Datos de entrenamiento y validación para predecir la densidad poblacional de la colmena.

Elaborado por: El investigador.

3.9.5 Creación del modelo de red neuronal.

La construcción del modelo se muestra en la Figura 3.40 y consta de la siguientes características: una capa LSTM oculta con cinco celdas de memoria y una capa de salida "*dense*", la función *return_sequences:* permite conectar varias capas LSTM seguidas, si la siguiente capa no es una capa LSTM se debe setear en "*False*", la capa de entrada LSTM () se especifica mediante el argumento "*input_shape*" en la primera capa oculta de la red y se requiere especificar el numero de muestras "*Samples*", la cantidad de pasos de tiempo "*Time Steps,*" y la cantidad de características "*Features*", su función de activación no lineal es tangente hiperbólica "*Tahn*" que genera valores entre -1.0 y 1.0, el algoritmo de optimización es "*adam*" el cual es una extensión del descenso de gradiente estocástico para actualizar los pesos de la red de forma iterativa en función de los datos de entrenamiento, para medir el rendimiento del modelo se utilizo la función del error cuadrático medio "*mean_squared_error*" el cual calcula el promedio de las diferencias cuadráticas entre los valores predichos y reales.

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout, LSTM

model=Sequential()
model.add(LSTM(5, return_sequences=False,␣
 ↪input_shape=(X_train.shape[1],1), activation='tanh'))
model.add(Dense(1))
model.compile(optimizer='adam',␣
 ↪loss='mean_squared_error')
```

```
model.fit(X_train, Y_train, epochs=100, batch_size=5,␣
 ↪validation_data=(X_test, Y_test))
```

Figura 3.40: Construcción del modelo de red neuronal artificial.
Elaborado por: El investigador.

La función *model.fit()* evalúa el modelo usándolo para hacer predicciones en el conjunto de datos de entrenamiento (*X_train, Y_train*) y luego compara las predicciones (*X_test, Y_test*).
La función *batch_size* es un hiperparámetro que define el número de muestras para trabajar antes de actualizar los parámetros internos del modelo.
La función *epochs* es un hiperparámetro que define el número de veces que el algoritmo de aprendizaje funcionará en todo el conjunto de datos de entrenamiento.

La configuración del modelo consta de 120 muestras con un tamaño de lote de cinco ("*batch_size=5*") y 100 épocas ("*epochs=100*"), esto quiere decir que el conjunto de datos se dividirá en 24 lotes, cada uno con cinco muestras, los pesos se actualizaran después de cada lote de cinco muestras, esto significa que una época implicará 24 lotes o 24 actualizaciones del modelo, con 100 épocas el modelo pasará por todo el conjunto de datos 100 veces para un total de 2400 lotes durante todo el proceso de entrenamiento.

Después de entrenar el modelo de red durante 100 épocas, el modelo intenta aprender el patrón y el comportamiento de los datos, puesto que se divide en conjuntos de entrenamiento y validación, se puede predecir el valor de los datos de prueba y

compararlos con los datos verdaderos.

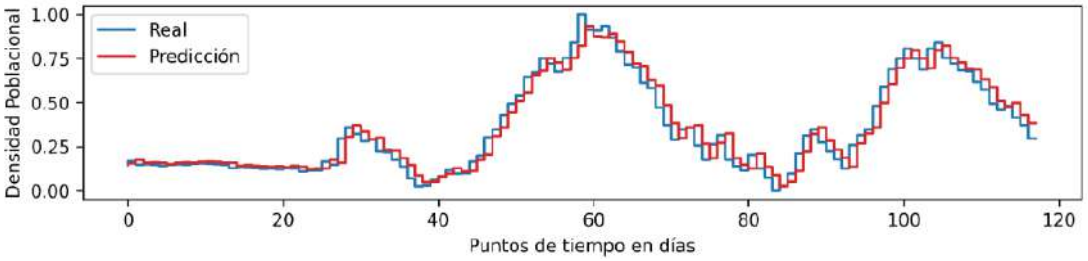


Figura 3.41: Entrenamiento y validación de datos de la densidad de población apícola.
Elaborado por: El investigador.

Como se puede observar en la Figura 3.41 , la arquitectura de la red no es bueno, básicamente está repitiendo los valores anteriores y posee ligeros cambio en la fluctuación de los datos de predicción, por lo tanto el modelo no puede predecir el futuro a partir del los valores anteriores, de modo que al ajustar el modelo cambiando el numero de capas ocultas, numero de neuronas y el numero de épocas de la red hasta que la fluctuación del error sea mínima para la predicción.

3.9.6 Definir y ajustar el modelo

Se presentan los resultados de cinco pruebas para entrenar (*Train*) y validar (*Test*) la red neuronal, cambiando el numero de neuronas en la capa oculta (*Neuronas*), el tamaño del lote (*batch_size*), el número de épocas (*epochs*) y cambiando la función de activación (*activation=”*), de este modo comparar el desempeño de la red y seleccionar cual se ajusta al modelo de predicción. En la Figura 3.42 se aprecia los parámetros a modificar para definir y ajustar el modelo

```
model=Sequential()
model.add(LSTM(N, return_sequences=False,␣
 ↪input_shape=(X_train.shape[1],1), activation='tanh'))
#model.add(Dropout(.2))
model.add(Dense(1))
model.compile(optimizer='adam',␣
 ↪loss='mean_squared_error')
model.fit(X_train, Y_train, epochs=?, batch_size=?,␣
 ↪validation_data=(X_test, Y_test))
```

Figura 3.42: Estructura del modelo de red neuronal.
Elaborado por: El investigador.

Para evaluar el modelo la métrica del coeficiente de determinación R^2 (o R al cuadrado) determina la calidad del modelo para replicar resultados, proporciona el porcentaje de la variación de la variable de respuesta comprendida entre 0 y 100 % e influyen en cómo se valora la importancia de las diferentes configuraciones en las pruebas de entrenamiento (*% Train*) y validación (*% Test*) y su elección final depende del porcentaje que mejor se ajustare el modelo a los datos, en la Figura 3.43 se muestra como medir el coeficiente de determinación para el entrenamiento y validación de los datos.

% Entrenamiento:

```
from sklearn.metrics import r2_score
Y_pred = model.predict(X_train)
print('R-Squared: %f'%(r2_score(Y_pred, Y_train)))
```

% Validación:

```
from sklearn.metrics import r2_score
Y_pred = model.predict(X_test)
print('R-Squared: %f'%(r2_score(Y_pred, Y_test)))
```

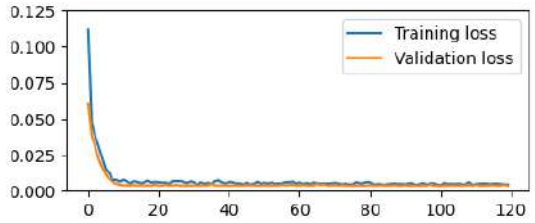
Figura 3.43: Coeficiente de determinación R^2 para el entrenamiento y validación.
Elaborado por: El investigador.

Tabla 3.19: Resultados de las pruebas hechas para diferentes configuraciones.

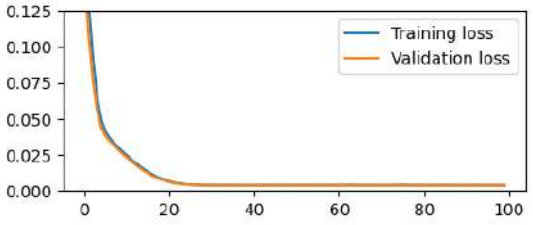
# Pruebas		# Neuronas	% de Ent	% de Val
1	**batch_size=1** **epochs=120** **Dropout=20 %**	1	92.93	90.74
		5	94.53	92.75
		10	94.26	93.16
		15	95.07	93.02
		25	95.18	94.51
		35	95.59	93.34
2	**batch_size=5** **epochs=100** **activation='tanh'**	1	42.81	-9.4
		5	89.09	87.86
		10	89.82	88
		15	92.11	91.19
		25	92.84	92.96
		35	93.02	91.36
3	**batch_size=10** **epochs=150** **activation='tanh'**	1	19.57	-98.16
		5	89.47	86.49
		10	89.16	87.79
		15	89.06	89.17
		25	90.16	89.67
		35	89.97	89.54
4	**batch_size=15** **epochs=120** **Dropout=20 %**	1	-33.75	-66.77
		5	84.89	84.9
		10	84.47	80.21
		15	91.08	90.72
		25	86.54	84.59
		35	89.39	87.85
5	**batch_size=30** **epochs=100** **activation='tanh'**	1	80.61	55.84
		5	70.25	11.18
		10	88.24	82.37
		15	83.47	73.47
		25	87.85	85.75
		35	84.44	78.97

Elaborado por: El investigador.

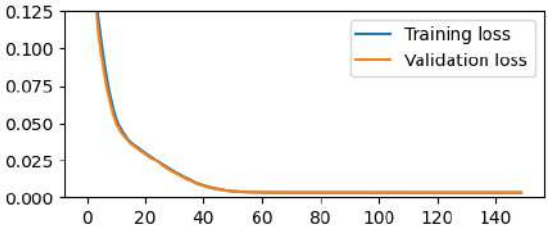
Para la selección del modelo a implementar en el prototipo se tomó en cuenta el porcentaje de entrenamiento y validación, por lo que la selección más apropiada como se puede observar en el la Tabla 3.19, el valor del coeficiente de determinación R^2 con mayor porcentaje de aciertos en la predicción con 95.18 % en el entrenamiento y 94.51 % en la validación.



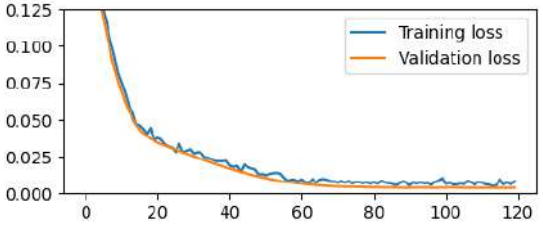
(a) Prueba 1, batch_size=1 y epochs=120.



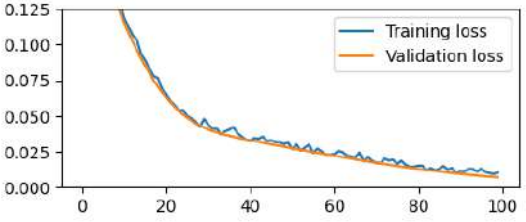
(b) Prueba 2, batch_size=5 y epochs=100.



(c) Prueba 3, batch_size=10 y epochs=150.



(d) Prueba 4, batch_size=15 y epochs=120.



(e) Prueba 5, batch_size=30 y epochs=100.

Figura 3.44: Fluctuación del error durante el entrenamiento.
Elaborado por: El investigador.

Analizando la Figura 3.44 el comportamiento del error durante el entrenamiento se puede observar en la Figura 3.44a muestra una mayor estabilidad, de ésta manera se determinó la mejor configuración para el modelo de red neuronal formándose con dos capa LSTM ocultas con 25 celdas de memoria, una regularización del 20 % (*model.add(Dropout(.2))* para reducir el sobreajuste y mejorar el error de generalización en redes neuronales profundas, una capa de salida (*model.add(Dense(1))*), el tamaño de lote es uno ("*batch_size=1*") y el numero épocas es de 120 ("*epochs=120*"), esto significa que una época pasara por todo el conjunto de datos 120 veces para un total de 14400 lotes durante todo el proceso de entrenamiento.

Finalmente, se puede generar predicciones utilizando la configuración anterior del modelo, para obtener una indicación visual de la habilidad del modelo, comparando

los datos reales y la predicción del conjunto de datos de la densidad poblacional de la colmena.

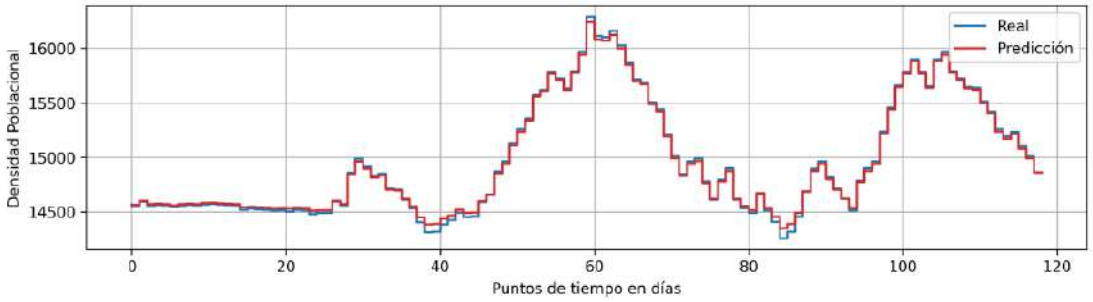


Figura 3.45: Comparación entre valores reales y la predicción de la densidad poblacional apícola.

Elaborado por: El investigador.

En la Figura 3.45 se puede observar que el modelo ha aprendido a imitar los datos y no tiene el retraso que solía tener con una simple capa LSTM oculta con una celda de memoria como se aprecia en la Figura 3.41, aunque todavía se está subestimando algunas observaciones en ciertas cantidades pero definitivamente el margen ha mejora en este modelo, cambiando el conjunto de entrada del modelo se sigue el mismo procedimiento para entrenar y validar los datos de: temperatura, humedad, dióxido de carbono y peso de la colmena, el código fuente se indica en el anexo 11.

Para validar el funcionamiento del modelo con la configuración previamente seleccionada se realizó el pronostico de datos como prueba final utilizando una submuestra de 31 días seguidos (1 al 31 de marzo) y una etapa de validación para la proyección de resultados utilizando los días del mes de abril.

3.9.7 Pronóstico de datos.

Debido a que la red neuronal se encuentra entrenada y los valores de validación son aceptables, se procede a obtener una nueva predicción, en cuyo caso, se utilizo los datos del mes de marzo para pronosticar la densidad poblacional, temperatura, humedad, dióxido de carbono y el peso de la colmena para el mes de abril.

Se preparan los datos para la validación, siguiendo el mismo procedimiento para el entrenamiento de datos, se enmarca como un problema de aprendizaje supervisado, se normaliza las variables de entrada y lo transforma en un problema del tipo supervisado como se muestra en las Figuras 3.46 y 3.47.

```
values = ultimos_Dias.values
values = values.astype('float32')
# normalizamos
values=values.reshape(-1, 1)
scaled = scaler.fit_transform(values)
reframed = series_to_supervised(scaled, PASOS, 1)
reframed.drop(reframed.columns[[7]], axis=1,␣
 ↪inplace=True)
reframed.head(7)
```

Figura 3.46: Normalización de los datos del pronostico.
Elaborado por: El investigador.

	var1(t-7)	var1(t-6)	var1(t-5)	var1(t-4)	var1(t-3)	var1(t-2)	var1(t-1)	var1(t)
7	-0.707821	-0.694048	-0.709788	-0.666502	-0.704869	-0.698967	-0.701918	-0.719625
8	-0.694048	-0.709788	-0.666502	-0.704869	-0.698967	-0.701918	-0.719625	-0.704869
9	-0.709788	-0.666502	-0.704869	-0.698967	-0.701918	-0.719625	-0.704869	-0.697983
10	-0.666502	-0.704869	-0.698967	-0.701918	-0.719625	-0.704869	-0.697983	-0.701918
11	-0.704869	-0.698967	-0.701918	-0.719625	-0.704869	-0.697983	-0.701918	-0.693064

Figura 3.47: Datos de entrada y salida del pronostico de datos.
Elaborado por: El investigador.

Del conjunto de datos “ultimosDías” se crea una función para predecir un nuevo valor, lo escala y normaliza la real, con el propósito de predecir los 30 días de abril, usando el conjunto de datos del mes de marzo, pronosticar la densidad poblacional, temperatura, humedad, dióxido de carbono y el peso de la colmena, según el resultado obtenido el apicultor puede tomar medidas preventivas ante situaciones donde los picos de la densidad poblacional cae drásticamente.

```
def nnuevoValor(x_test,nuevoValor):
    for i in range(x_test.shape[2]-1):
        x_test[0][0][i] = x_test[0][0][i+1]
    x_test[0][0][x_test.shape[2]-1]=nuevoValor
    return x_test

results=[]
for i in range(30):
    parcial=model.predict(x_test)
    results.append(parcial[0])
    print(x_test)
    x_test=agregarNuevoValor(x_test,parcial[0])

adimen = [x for x in results]
inverted = scaler.inverse_transform(adimen)
```

Figura 3.48: Pronostico de datos del mes de abril 2020
Elaborado por: El investigador.

Densidad poblacional de la colmena

En la Figura 3.49 se observa la salida de los datos reales y predicción de la densidad de población apícola del mes de abril, el modelo ha aprendido a imitar los datos aunque todavía se subestiman algunas observaciones, el modelo todavía puede ser mejorado, recolectando mas datos de los meses siguientes para entrenar y validar los datos hasta obtener una predicción mas precisa.

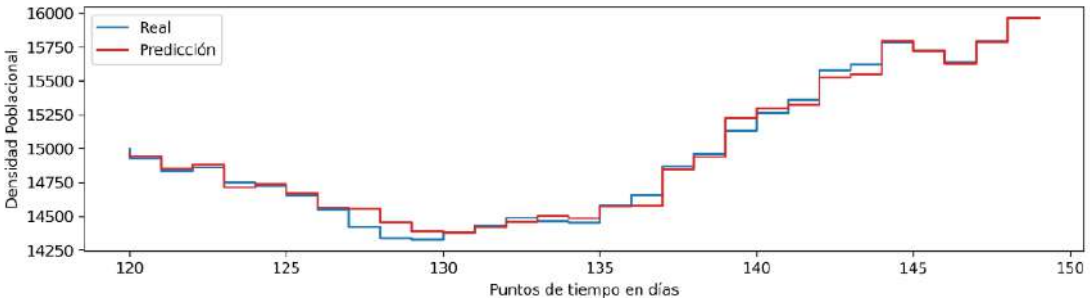


Figura 3.49: Comparación entre valores reales y la predicción de la población apícola para el mes de abril 2020.
Elaborado por: El investigador.

Peso de la colmena

La Figura 3.50 muestra la salida de los datos reales y predicción del peso de la colmena del mes de abrir, excepto por algunas observaciones el sistema seria perfecto, el modelo todavía necesita un mayor numero de datos de entrenamiento para validar la salida con

mayor precisión.

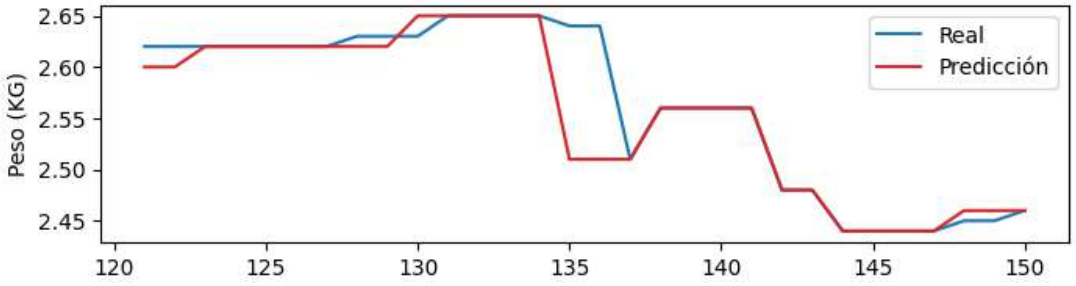
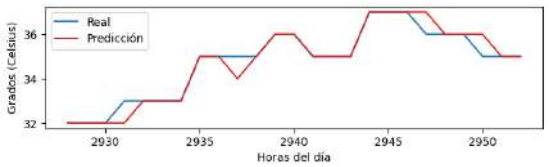


Figura 3.50: Comparación entre valores reales y la predicción del peso de la colmena para el mes de abril 2020.

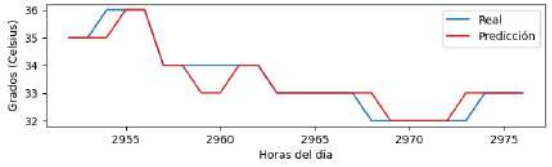
Elaborado por: El investigador.

Temperatura de la colmena

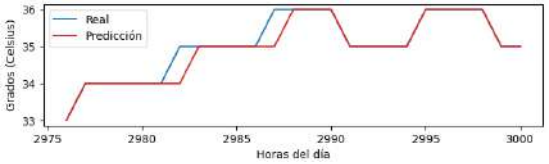
Los datos de entrenamiento y validación de la temperatura de la colmena es mayor a los datos de la densidad de población apícola, por ende los valores de predicción son mas exactos a los valores reales, ya que posee un mayor numero de datos para entrenar al modelo de la red y validar su salida con mayor precisión, en la Figura 3.51 se observa la salida de los datos reales y perdición de la primera semana del mes de abril, el pronostico es aceptable y se mantiene dentro del rango óptimo de temperatura de la colmena.



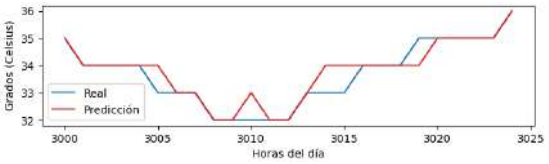
(a) Valores reales y predicción para el 2020-04-01



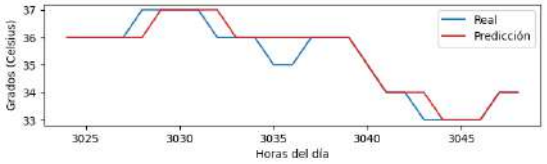
(b) Valores reales y predicción para el 2020-04-02



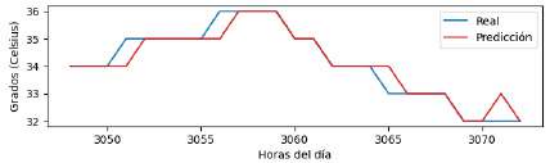
(c) Valores reales y predicción para el 2020-04-03



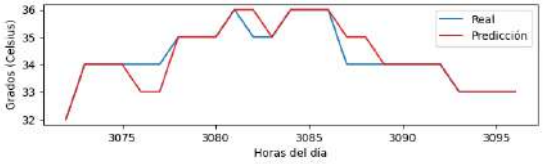
(d) Valores reales y predicción para el 2020-04-04



(e) Valores reales y predicción para el 2020-04-05



(f) Valores reales y predicción para el 2020-04-06

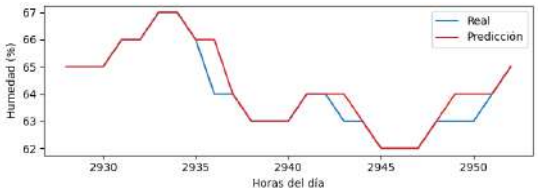


(g) Valores reales y predicción para el 2020-04-07

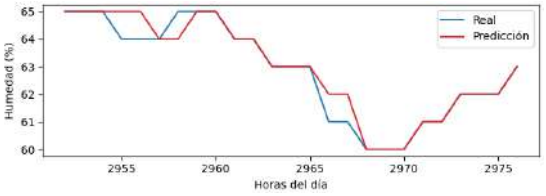
Figura 3.51: Comparación entre valores reales y la predicción de la temperatura para la primera semana del mes de abril 2020

Humedad de la colmena

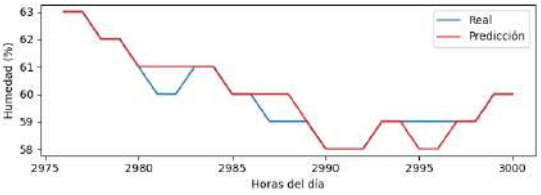
Los datos reales y predicción de la Figura 3.52 muestra los valores de la humedad relativa dentro de la colmena de la primera semana del mes de abril, a pesar de tener valores que ascienden y descienden bruscamente en ciertas horas del día el sistema los predice con pequeñas alteraciones, a pesar de ello la predicción es aceptable.



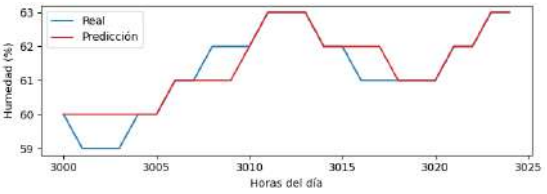
(a) Valores reales y predicción para el 2020-04-01



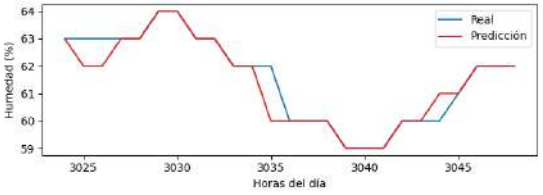
(b) Valores reales y predicción para el 2020-04-02



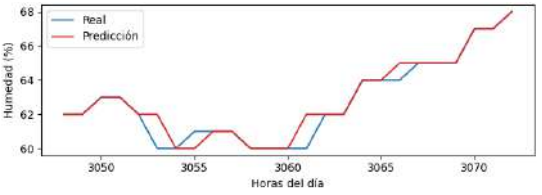
(c) Valores reales y predicción para el 2020-04-03



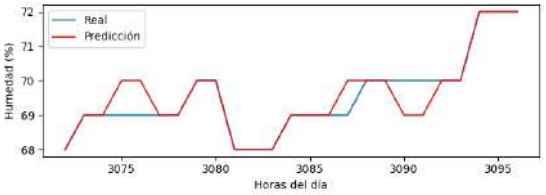
(d) Valores reales y predicción para el 2020-04-04



(e) Valores reales y predicción para el 2020-04-05



(f) Valores reales y predicción para el 2020-04-06

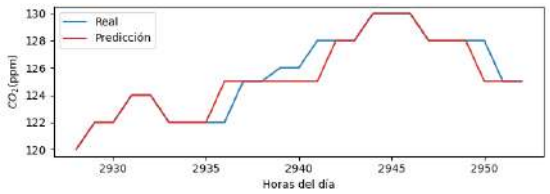


(g) Valores reales y predicción para el 2020-04-07

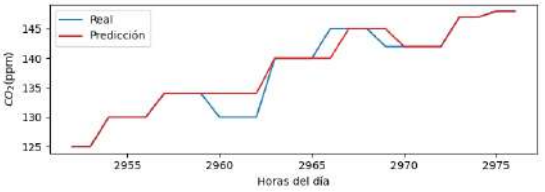
Figura 3.52: Comparación entre valores reales y la predicción de la humedad para la primera semana del mes de abril 2020

Dióxido de carbono dentro de la colmena

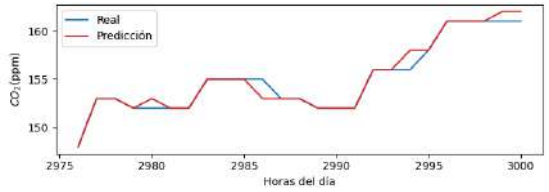
En la Figura 3.53 se observan los datos reales y predicción de los niveles de dióxido de carbono dentro de la colmena para la primera semana del mes de abril, el modelo imita los valores reales con una mínima variación en ciertas horas del día, aun así la predicción de los datos es aceptable.



(a) Valores reales y predicción para el 2020-04-01

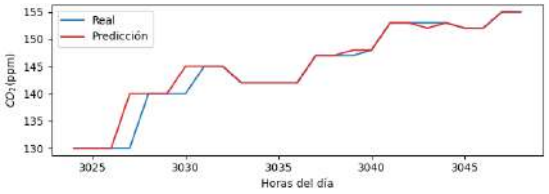


(b) Valores reales y predicción para el 2020-04-02

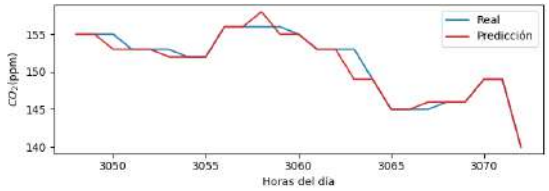


(c) Valores reales y predicción para el 2020-04-03

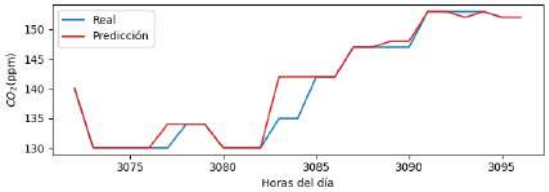
(d) Valores reales y predicción para el 2020-04-04



(e) Valores reales y predicción para el 2020-04-05



(f) Valores reales y predicción para el 2020-04-06



(g) Valores reales y predicción para el 2020-04-07

Figura 3.53: Comparación entre valores reales y la predicción de CO2 para la primera semana del mes de abril 2020

De la Figura3.53 se puede obtener la información de la Tabla 3.20 y 3.21 en donde se presentan los resultados de acierto y error en las predicciones del mes de abrir, definido por el modelo de la red previamente configurado y validado el conjunto de datos de la entrada de la red neuronal.

Tabla 3.20: Resultados en la predicción de la población apícola y peso de la colmena durante las pruebas finales.

DÍA	**Población**		**Peso**	
	% de acierto	% de error	% de acierto	% de error
2020-04-01 2020-04-15	93.06	6.94	82,32	17,68
2020-04-16 2020-04-30	92.20	7.8	90.2	9.8
TOTAL	92.63	7.37	86,26	13,74

Elaborado por: El investigador.

Tabla 3.21: Resultados en la predicción de temperatura, humedad y co2 durante las pruebas finales.

DÍA	**Temperatura**		**Humedad**		**CO2**	
	% de acierto	% de error	% de acierto	% de error	% de acierto	% de error
2020-04-01	83.33	16.67	70.83	29.17	89.01	10.99
2020-04-02	79.17	20.83	79.17	20.83	81.52	18.48
2020-04-03	91.67	8.33	75.00	25.00	93.55	6.45
2020-04-04	79.17	20.83	70.83	29.17	93.23	6.77
2020-04-05	79.17	20.83	70.83	29.17	77.00	23.00
2020-04-06	83.33	16.67	58.33	41.67	89.39	10.61
2020-04-07	79.17	20.83	79.17	20.83	77.92	22.08
TOTAL	82.14	17.86	72.02	27.98	85.94	14.06

Elaborado por: El investigado

Como se puede apreciar en la Tabla 3.20 el porcentaje promedio de acierto para la densidad de población apícola del mes de abril muestran un alto grado de asertividad en las predicciones, con un total del 92.63 %, manteniendo los valores esperados durante la mayoría de los días de prueba. Para la predicción del peso de la colmena del mes de abril muestra un descenso del porcentaje de acierto, con un total del 86.26 %, aún así, el resultado es aceptable.

Para el pronostico de la temperatura, humedad y dióxido de carbono de la tabla 3.21 el porcentaje promedio de acierto es del 82.14 %, 72,02 y 85,94 respectivamente, manteniendo los valores esperados durante la mayoría de días de prueba, se muestra un alto grado de asertivida, a excepción de la humedad que desciende de manera

considerable por debajo del 80 %, debido a la irregularidad de los datos registrados del mes de abril.

3.10 Interfaz de usuario.

Se desarrolló una página web con el framework Flask en donde el usuario puede visualizar los datos de los sensores de: temperatura, humedad, dióxido de carbono, densidad poblacional y peso de la colmena en tiempo real, utilizando el lenguaje de programación Python, facilita la conexión entre la página web y la base de datos, se configura de manera rápida y eficiente con pocas lineas de código, facilitando la compilación de la interfaz de la aplicación web.

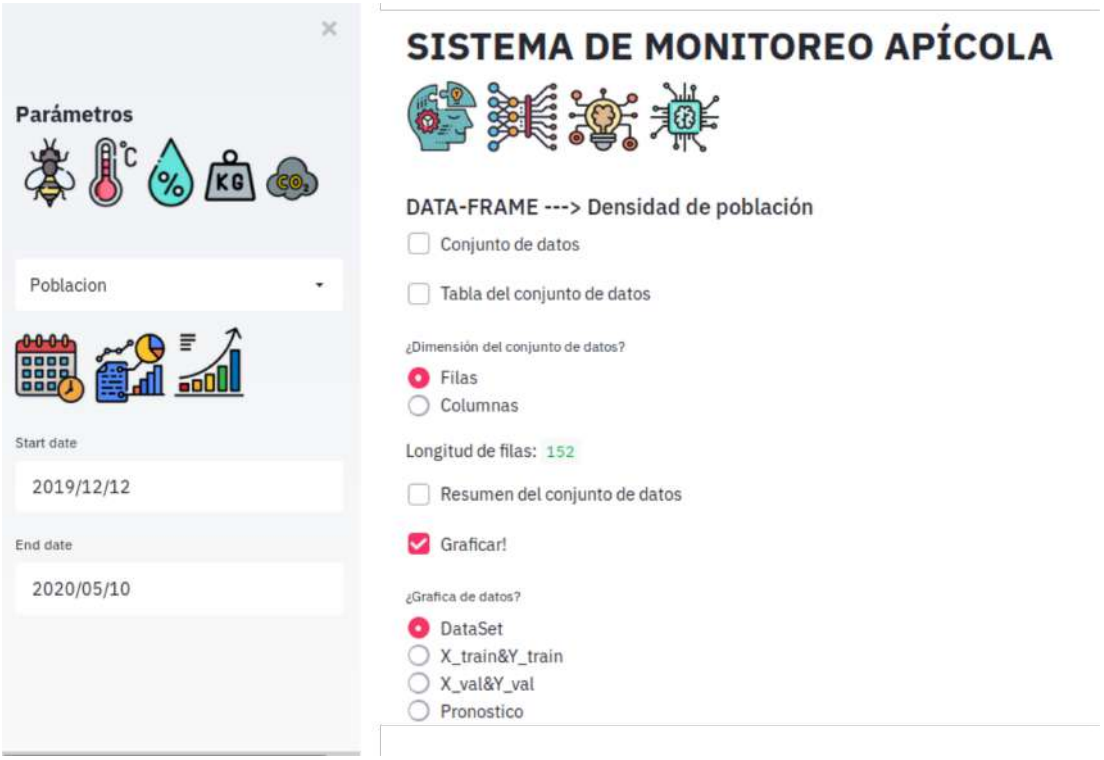


Figura 3.54: Interfaz de usuario web.
Elaborado por: El investigado

Al ejecutar el servidor web Flask genera un dominio por defecto con una URL (http://127.0.0.1:5000/), al ingresar el dominio en el navegador se observa la pagina principal como se muestra en la Figura 3.54, donde se puede visualizar los datos registrados por los sensores, además se puede editar e insertar datos en caso de ser

necesario en la base de datos MySql. Para entrenar y validar el modelo de la red neuronal se extrae de la base de datos el "DataSet.csv" de los sensores en formato csv por su fácil manipulación con las librerías de Python, al cargar el conjunto de datos de los sensores la aplicación web carga la configuración del modelo y visualiza los detalles de la red neuronal tales como: el numero de neuronas, capas ocultas, capas de salida, función de activación, el tipo de optimizador, numero de datos de entrenamiento y validación y sus respectivas gráficas de entrenamiento, validación y predicción del modelo, el código fuente de la aplicación se indica en el anexo 12.

3.11 Presupuesto

El presente apartado analiza los costos de la implementación del sistema de monitorización por medio de tecnología inalámbrica para la predicción de la variación de población apícola. El costos del diseño va ligado al salario básico de un Ingeniero en Electrónica y Comunicaciones en el país, el cual está determinado por el Ministerio de Trabajo, por una valor de $858,00 al mes [58].
Se considera un promedio de 25 días laborales al año, el salario diario esta dado por:

$$Salario\_diario = \frac{Salario\_mensual}{Das\_laborables}$$
$$Salario\_diario = \frac{858,00}{25}$$
$$Salario\_diario = \$34,32$$

Para obtener la remuneración por hora de trabajo es necesario realizar el siguiente cálculo:

$$Salarioporhora = \frac{Salario\_diario}{Horas\_laborables}$$
$$Salario\_por\_hora = \frac{34,32}{8}$$
$$Salario\_por\_hora = \$4,29$$

De acuerdo a las horas empleadas para el diseño del proyecto, incluyendo realización

de simulaciones, y pruebas de funcionamiento, el costo de diseño está dado por.

$$Costo\_de\_diseo = Horas\_de\_diseo * Salario\_por\_hora$$

$$Costo\_de\_diseo = 112 * 4,29$$

$$Costo\_de\_diseo = 480,48$$

En la Tabla 3.22 se muestra el valor de los elementos empleados para la construcción del prototipo.

Tabla 3.22: Presupuesto para la construcción del sistema de monitoreo.

PRESUPUESTO					
Item	**Descripción**	**Unidad**	**Cantidad**	**Costo unitario $**	**Costo total $**
1	DHT21	c/u	1	4	4
2	MQT135	c/u	1	4.5	4.5
3	HX711	c/u	1	5	5
4	Celda de carga	c/u	4	4	20
5	Arduino Mega	c/u	1	15	15
6	ESP-01	c/u	1	4	4
7	Batería	c/u	1	8	8
8	Resistencias	c/u	16	0.10	1.60
9	Capacitor 101	c/u	16	0.10	1.60
10	Elaboración de la placa	c/u	1	10	10
11	Elaboración de la caja	c/u	1	60	60
12	Estaño	c/u	1	1.50	1.50
13	Borneras	c/u	5	0.50	2.50
14	Espadines	c/u	3	0.50	1.50
				Subtotal	139.2
				IVA (12 %)	16,70
				Total	155,9
				Imprevistos (3 %)	4,68
				Total	160,58

Elaborado por: El investigador

Por último, para conocer el costo total de implementación del prototipo se considerar la siguiente tabla:

Tabla 3.23: Presupuesto total del diseño del sistema de monitoreo.

N°	Descripción	Unidad	Cantidad	Valor Unitario	Valor Total
1	Costo de diseño	c/u	1	480,48	480,48
2	Costo de construcción	c/u	1	160,58	160,58
TOTAL					641,06

Elaborado por: El investigador

En la Tabla 3.23 se observa que el costo más elevado es el del diseño del prototipo del sistema de monitoreo apícola, sin embargo al considerarse en producción en línea el valor disminuirá ya que se dispondrá del diseño y bastará con la construcción del sistema.

Capítulo IV

CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones

- Se ha diseñado el sistema de monitoreo apícola de bajo costo para medir las variables de temperatura, humedad, dióxido de carbono, peso y la densidad poblacional dentro de la colmena de abejas melíferas, el sistema reduce la intervención del hombre sobre la colmena, evitando posibles daños en su ecosistema, ofrece información en tiempo real sobre algún cambio significativo en su entorno que afecten directamente el rendimiento y producción de la colmena, favoreciendo al apicultor al tomar medidas preventivas para mejorar el rendimiento de la colmena.
- La implementación del sistema de monitoreo apícola se propone a predecir la variación de población de abejas melíferas basado en la arquitectura artificial de red neuronal recurrente de memoria a corto y largo plazo (LSTM), aprenden los comportamientos pasados mas importantes y comprenden si esos comportamientos son características importantes para hacer predicciones futuras, los resultados obtenidos en la predicción de la densidad poblacional de la colmena, confirman la aplicabilidad del modelo de la red con una asertividad mayor al 90 %, así se muestra la viabilidad en la implementación del sistema.
- El uso de las bibliotecas del aprendizaje profundo Keras y Tensorflow, permite construir redes profundas sin comprender completamente el funcionamiento interno de una red neuronal, construidos sobre bibliotecas matemáticas simbólicas que permite el calculo de los pesos de la neurona en cada capa de la red de forma rápida y eficiente, minimizando el error durante el entrenamiento y validación del modelo para que se generalice bien y no se sobreajuste o se sesgue los datos del entrenamiento para evitar predicciones erróneas futuras.

- Las predicciones de las variables de temperatura, humedad, dióxido de carbono, densidad poblacional y peso de la colmena, cambian según el tamaño del lote (batch_size) limitando el numero de muestras de entrenamiento para trabajar antes de que se actualicen los parámetros internos del modelo, el numero de épocas (epoch) afecta el comportamiento dinámico del modelo y el número de neuronas afecta la capacidad de aprendizaje de la red, entre más neuronas mayor tiempo de entrenamiento creando un sobre-ajuste de los datos de capacitación, ajustar las redes neuronales es un trabajo empírico difícil, sebe diseñar un conjunto de pruebas para evaluar la configuración de la red LSTM para el pronóstico de series temporales.

4.2 Recomendaciones

- Para la implementación del prototipo del sistema de monitorio apícola en el campo de prueba se recomienda mejorar la estabilidad del sensor capacitivo cambiando los valores resistivos y capacitivos para la detección de presencia en entrada/salida de las abejas ya que los valores arbitrarios emitidos por el sensor son altos y limita la detección en intervalos de tiempo cortos.
- Se debe tener cuidado con el sobre-ajuste de datos, esto ocurre cuando una red neuronal esencialmente memoriza los datos de entrenamiento, el modelo de la red se vuelve inútil en la predicción fuera de la muestra, por tal motivo se especificar el tamaño del lote (batch_size) y el número de épocas (epoch) para el algoritmo de aprendizaje profundo.
- El tamaño del lote debe ser mayor o igual que uno y menor o igual que el número de muestras en el conjunto de datos de entrenamiento y el número de épocas se puede establecer en un valor entero entre uno e infinito, hasta que el error del modelo se haya minimizado lo suficiente a lo largo del tiempo.
- Para evaluar el rendimiento del modelo de la red se recomienda tener un conjunto de datos mayor a un año completo, para poder afirmar que el prototipo trabaje de forma efectiva durante un periodo prolongado, el tamaño de la muestra se debe separar en una porción del 80/20 % para entrenar y validar el modelo.

- Dependiendo del tamaño del conjunto de datos, los modelos basados en el aprendizaje profundo pueden tomar horas, días e incluso semanas para entrenar, es recomendable guardar la arquitectura y los pesos asociados a la red del modelo, Keras proporciona la facilidad de almacenar grandes cantidades de datos numéricos en formato hdf5.

Bibliografía

[1] G. Kritsky, *The Quest for the Perfect Hive: A History of Innovation in Bee Culture*, February 24th 2010.

[2] H. Blackiston, *Beekeeping For Dummies*, 4th ed. John Wiley & Sons, Inc., 2017.

[3] F. Chollet, *Deep Learning with Python*, 1st ed. USA: Manning Publications Co., 2017.

[4] D. Sarkar, R. Bali, and T. Sharma, *Practical Machine Learning with Python*, 2018.

[5] N. K. Manaswi, *Deep Learning with Applications Using Python*, 2018.

[6] P. Goyal, S. Pandey, and K. Jain, Eds., *Deep Learning for Natural Language Processing Creating Neural Networks with Python*. Apress, 2018.

[7] Naylamp-Mechatronics, "Sensor calidad aire mq135," naylamp mechatronics," [En Linea.] https://naylampmechatronics.com/sensores-gas/73-sensor-calidad-aire-mq135.html.

[8] N. Mechatronics, "Módulo hx711 transmisor de celda de carga," [En Linea.] https://naylampmechatronics.com/sensores/147-modulo-hx711-transmisor-de-celda-de-carga.html.

[9] Arduino, "Arduino mega 2560 rev3 | arduino official store," [En Linea.]https://store.arduino.cc/usa/mega-2560-r3.

[10] TuVoltio, "Bateria seca 12v 5a - tuvoltio," [En Linea.]https://www.tuvoltio.com/categorias/77-accesorios-varios/1883-bateria - seca-12v-5a.

[11] Zaragoza-Makerspace, "Programar el esp-01 mediante un nodemcu - zaragoza makerspace," [En

Linea.]https://zaragozamakerspace.com/index.php/programar-el-esp-01-mediante-un-nodemcu/.

[12] P. Medrzycki, F. Sgolastra, L. Bortolotti, G. Bogo, S. Tosi, E. Padovani, C. Porrini, and A. G. Sabatini, "Influence of brood rearing temperature on honey bee development and susceptibility to poisoning by pesticides," *Journal of Apicultural Research*, vol. 49, no. 1, pp. 52–59, 2010. [Online]. Available: https://doi.org/10.3896/IBRA.1.49.1.07

[13] A. Kviesis and A. Zacepins, "Application of neural networks for honey bee colony state identification," in *2016 17th International Carpathian Control Conference (ICCC)*, 2016, pp. 413–417.

[14] V. G. Rybin, D. N. Butusov, T. I. Karimov, D. A. Belkin, and M. N. Kozak, "Embedded data acquisition system for beehive monitoring," pp. 387–390, Oct 2017.

[15] A. Zacepins, A. Kviesis, A. Pecka, and V. Osadcuks, "Development of internet of things concept for precision beekeeping," pp. 23–27, May 2017.

[16] Agrimundo, "Inteligencia competitiva para el sector agroalimentario, situación mundial del síndrome de colapso de las abejas,," [En Linea.] www.agrimundo.gob.cl/wp-ontent/uploads/130826_reporte_apicultura_n22.pdf , 2013.

[17] Greenpeace, "El declive de las abejas, peligros para los polinizadores y la agricultura de europas," [En Linea.] http://archivoes.greenpeace.org/espana/Global/espana/report/Agricultura-ecologica/el_declive_de_las_abejas.pdf , 2013.

[18] El-Telégrafo, "Reporte apícola en el ecuador," [En Linea.] https://www.eltelegrafo.com.ec/noticias/economia/4/ecuador-productores-mielapicultura, 2018.

[19] MAG, "Ministerio de agricultura y ganadería, registro de apicultores en el ecuador," [En Linea.] https://www.agricultura.gob.ec/ecuador-tiene-1760-

apicultores-registrados/, 2018.

[20] El-Telégrafo, "La apicultura rinde como alternativa de producción," [En Linea.] https://www.eltelegrafo.com.ec/noticias/economia/4/la-apicultura-rindecomo-alternativa-de-produccion, 2016.

[21] R. Gupta, W. Reybroeck, J. van Veen, and A. Gupta, Eds., *Beekeeping for poverty alleviation and livelihood security: Vol.1: Technological aspects of beekeeping*. Springer Science+Business Media, 2014.

[22] H. Smith, Ed., *MACHINE LEARNING FOR BEGINNERS The Absolute Beginner's Guide to Learn and Understand Machine Learning Effectively*, 2019.

[23] M. Kirk, *Thoughtful Machine Learning with Python*, 2017, vol. First Edition.

[24] A. Menshawy, *Deep Learning By Example*, 1st ed. Packt Publishing, Ltd., 2018.

[25] V. K. Ayyadevara, *Neural Networks with Keras Cookbook*. Packt Publishing Ltd., 2019.

[26] M. Taylor, *Neural Networks: A Visual Introduction For Beinners*. Blue Windmill Media, 2017.

[27] M. Ali, D. Jayakody, Y. Chursin, S. Affes, and S. Dmitry, "Recent advances and future directions on underwater wireless communications," *Archives of Computational Methods in Engineering*, Jan. 2019.

[28] EcuRed, "Red inalámbrica," [En Linea.]https://www.ecured.cu/Red_inal %C3 % A1mbrica.

[29] C. Olivares, "Modelo de cobertura para redes inalámbricas de interiores," [En Linea.]http://bibing.us.es/proyectos/abreproy/11761/fichero/Volumen1 %252F5 -Cap %C3 %ADtulo1+-+Introducci %C3 %B3n+a+las+redes+inal %C3 %A1mbricas.pdf+, 2009.

[30] C. Digitales, "El estándar bluetooth, ieee 802.15.1," [En Linea.]hhttp://catarina.udlap.mx/u_dl_a/tales/documentos/lem/archundia_p_fm/capitulo3.pdf.

[31] Universidad-Técnica-Federico-Santa-María, “Estudio del estándar zigbee,” [En Linea.]http://profesores.elo.utfsm.cl/ agv/elo322/1s18/projects/reports/Zigbee.pdf.

[32] Radiocomunicaciones, “Estándares inalámbricos,” [En Linea.] http://www.radiocomunicaciones.net/pdf/wifi/tabla-de-estandares-inalambricos.pdf.

[33] G. Parra, “Sistema de comunicaciÓn gprs para procesos operativos,” [En Linea.] https://docplayer.es/10089300-Sistema-de-comunicacion-gprs-para-procesos-operativos-gary-gustavo-parra-torralvo.html.

[34] P. R. Clericus, “Análisis y estudio de redes gprs,” [En Linea.]http://cybertesis.uach.cl/tesis/uach/2005/bmfcis211a/doc/bmfcis211a.pdf.

[35] R. digital de la Facultad de Ingeniería UNAM, “Microcontroladores,” [En Linea.]http://www.ptolomeo.unam.mx:8080/jspui/bitstream/132.248.52.100/760/4/A4.pdf.

[36] Microcontroladores, “Sensores con microcontroladores,” [En Linea.]https://conmicrocontroladores.com/sensores/.

[37] PCE-Ibérica-S.L., “Equipos de medida - balanzas - regulación y control,” [En Linea.]https://www.pce-iberica.es/instrumentos-de-medida/sistemas/.

[38] KEYENCE, “Fundamentos del sensor: Guía de sensores para fábricas clasificados por principios,” [En Linea.] https://www.keyence.com.mx/ss/products/sensor/sensorbasics/photoelectric/info/.

[39] y. . . m. . . p. . . t. . M. v. . . j. . U. d. . . Ángela Méndez, Octavio Márquez.

[40] M. Becher, H. Scharpenberg, and R. Moritz, “Pupal developmental temperature and behavioral specialization of honeybee workers (apis mellifera l.),” *Journal of comparative physiology. A, Neuroethology, sensory, neural, and behavioral physiology*, vol. 195, pp. 673–9, 05 2009.

[41] J. Jones, M. Myerscough, S. Graham, and B. Oldroyd, “Honey bee nest thermoregulation: Diversity promotes stability,” *Science (New York, N.Y.)*, vol. 305, pp. 402–4, 08 2004.

[42] P. Medrzycki, F. Sgolastra, L. Bortolotti, G. Bogo, S. Tosi, E. Padovani, C. Porrini, and S. A.G., "Influence of brood rearing temperature on honey bee development and susceptibility to poisoning by pesticides," *Journal of Apicultural Research*, vol. 49, pp. 52–59, 01 2010.

[43] A. remote hive monitoring, "Hive humidity | arnia," [En Linea.] https://www.arnia.co.uk/hive-humidity/ , 2017.

[44] G. Seritan, B.-A. Enache, A. Florin, F. Adochiei, and S. Toader, "Low cost platform for monitoring honey production and bees health," 05 2018, pp. 1–4.

[45] Ecocolmena, "Curso avanzado de apicultura ecológica en girona," [En Linea.] https://ecocolmena.com/abejas-pasando-el-invierno-en-galpones/, 2017.

[46] J. Ivars, "La regla de farrar," [En Linea.] https://www.latiendadelapicultor.com/blog/regla-de-farrar/, 09-Jul-2015.

[47] C. L. Farrar, "The Evaluation of Bees for Pollination," *Journal of Economic Entomology*, vol. 24, no. 3, pp. 622–627, 06 1931. [Online]. Available: https://doi.org/10.1093/jee/24.3.622

[48] Meikle, William G., Rector, Brian G., Mercadier, Guy, and Holst, Niels, "Within-day variation in continuous hive weight data as a measure of honey bee colony activity," *Apidologie*, vol. 39, no. 6, pp. 694–707, 2008. [Online]. Available: https://doi.org/10.1051/apido:2008055

[49] Naylamp-Mechatronics., "Sensor de temperatura y humedad relativa dht22 (am2302) - naylamp mechatronics," [En Linea.] https://naylampmechatronics.com/sensores-temperatura-y-humedad/58-sensor-de-temperatura-y-humedad-relativa-dht22-am2302.html.

[50] Load sensor - 50kg - sen-10245 - sparkfun electronics. [Online]. Available: https://www.sparkfun.com/products/10245

[51] Sensor de fuerza o presión mf01 - hetpro/tutoriales. [Online]. Available: http://hetpro-store.com/TUTORIALES/sensor-de-fuerza-o-presion-mf01/

[52] Real decreto 209/2002, de 22 de febrero, por el que se establecen normas de ordenación de las explotaciones apícolas. [Online]. Available: https://boe.es/buscar/pdf/2002/BOE-A-2002-5016-consolidado.pdf

[53] J. B. Magem, "Colmena y portanúcleo tipo langstroth, informe técnico para la construcción de una colmena y portanúcleo tipo langstroth," *Universidad Nacional Agraria La Molina (UNALM)*, 2015.

[54] G. T. Library, "Flask vs django: Must know differences," [En Linea.]https://www.guru99.com/flask-vs-django.html.

[55] 2ndquadrant postgresql, "Postgresql vs mysql: advantage and disadvantages," [En Linea.]https://www.2ndquadrant.com/es/postgresql/postgresql-vs-mysql/.

[56] D. Cervan, "Guía básica para entender los algoritmos de machine learning," [En Linea.]https://dheybicervan.com/guia-basica-para-entender-los-algoritmos-de-machine-learning/.

[57] S. Kojouharov, "Cheat sheets for ai, neural networks, machine learning, deep learning and data science," [En Linea.]https://becominghuman.ai/cheat-sheets-for-ai-neural-networks-machine-learning-deep-learning-big-data-science-pdf-f22dc900d2d7.

[58] M. de Trabajo, "Salarios mínimos sectoriales," [En Linea.]http://www.trabajo.gob.ec/wp-content/uploads/2019/02/acuerdo-ministerial -Nro.-MDT-2019-008-A-1.pdf.

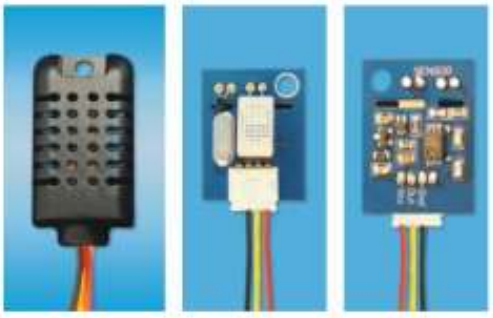
[59] J. S. S. C. y. A. P. Javier Usabiaga, José Gallardo, "Manual básico de apicultura," *SECRETARÍA DE AGRICULTURA, GANADERÍA, DESARROLLO RURAL, PESCA Y ALIMENTACIÓN*, pp. 8–16.

ANEXOS

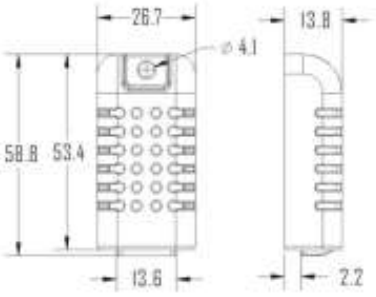
Anexos 1: HOJA DE DATOS TÉCNICOS DEL SENSOR DTH21

1、Product Overview

AM2301 capacitive humidity sensing digital temperature and humidity module is the one that contains the compound has been calibrated digital signal output of the temperature and humidity sensor. Application of a dedicated digital modules collection technology and the temperature and humidity sensing technology, to ensure that the product has high reliability and excellent long-term stability. The sensor includes a capacitive sensor wet components and a high-precision temperature measurement devices, and connected with a high-performance 8-bit microcontroller. The product has excellent quality, fast response, strong anti-jamming capability, and high cost. Each sensor is extremely accurate humidity calibration chamber calibration. The form of procedures, the calibration coefficients stored in the microcontroller, the sensor within the processing of the heartbeat to call these calibration coefficients. Standard single-bus interface, system integration quick and easy. Small size, low power consumption, signal transmission distance up to 20 meters, making it the best choice of all kinds of applications and even the most demanding applications. Products for the 3-lead (single-bus interface) connection convenience. Special packages according to user needs.



Physical map



Dimensions (unit: mm)

2、Applications

HVAC, dehumidifier, testing and inspection equipment, consumer goods, automotive, automatic control, data loggers, home appliances, humidity regulator, medical, weather stations, and other humidity measurement and control and so on.

3、Features

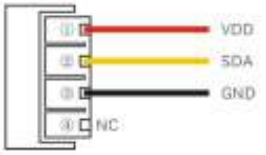
Ultra-low power, the transmission distance, fully automated calibration, the use of capacitive humidity sensor, completely interchangeable, standard digital single-bus output, excellent long-term stability, high accuracy temperature measurement devices.

4、The definition of single-bus interface

4.1 AM2301 Pin assignments

Table 1: AM2301 Pin assignments

Pin	Color	Name	Description
1	Red	VDD	Power (3.3V-5.2V)
2	Yellow	SDA	Serial data, Dual-port
3	Black	GND	Ground
4		NC	Empty



PIC1: AM2301 Pin Assignment

4.2 Power supply pins（VDD GND）

AM2301 supply voltage range 3.3V – 5.2V, recommended supply voltage is 5V.

4.3 Serial data（SDA）

SDA pin is tri structure for reading, writing sensor data. Specific communication timing, see the detailed description of the communication protocol.

5、Sensor performance

5.1 Relative humidity

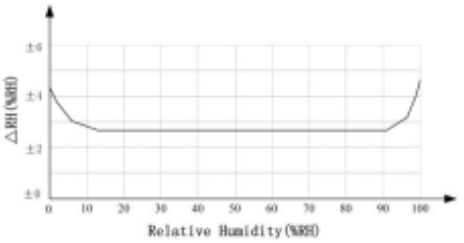
Table 2: AM2301 Relative humidity performance table

Parameter	Condition	min	typ	max	Unit
Resolution			0.1		%RH
Range		0		99.9	%RH
Accuracy[1]	25℃		± 3		%RH
Repeatability			± 1		%RH
Exchange		Completely interchangeable			
Response [2]	1/e(63%)		<6		S
Sluggish			±0.3		%RH
Drift [3]	Typical		<0.5		%RH/yr

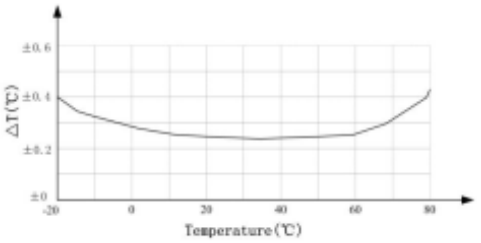
5.2 Temperature

Table 3: AM2301 Relative temperature performance

Parameter	Condition	min	typ	max	Unit
Resolution			0.1		℃
			16		bit
Accuracy			±0.3	± 1	℃
Range		−40		80	℃
Repeat			±0.2		℃
Exchange		Completely interchangeable			
Response	1/e(63%)		< 10		S
Drift			±0.3		℃/yr



Pic2: At25℃ The error of relative humidity



Pic3: The maximum temperature error

6、Electrical Characteristics

Electrical characteristics, such as energy consumption, high, low, input, output voltage, depending on the power supply. Table 4 details the electrical characteristics of the AM2301, if not identified, said supply voltage of 5V. To get the best results with the sensor, please design strictly in accordance with the conditions of design in Table 4.

Table 4; AM2301 DC Characteristics

Parameter	Condition	min	typ	max	Unit
Voltage		3.3	5	5.2	V
Power consumption [1]	Dormancy	10	15		μA
	Measuring		500		μA
	Average		300		μA
Low level output voltage	Ioc[5]	0		300	mV
High output voltage	Rp<25 kΩ	90%		100%	VDD
Low input voltage	Decline	0		30%	VDD
Input High Voltage	Rise	70%		100%	VDD
Rpu[6]	VDD = 5V VIN = VSS	30	45	60	kΩ
Output current	turn on		8		mA
	turn off	10	20		μA
Sampling period		2			S

[1] the accuracy of the factory inspection, the sensor 25 ° C and 5V, the accuracy specification of test conditions, it does not include hysteresis and nonlinearity, and is only suitable for non-condensing environment.

[2] to achieve an order of 63% of the time required under the conditions of 25 ℃ and 1m / s airflow.

[3] in the volatile organic compounds, the values may be higher. See the manual application to store information.

[4] this value at VDD = 5.0V when the temperature is 25 ℃, 25 / time, under the conditions of the average.

[5] low output current.

[6] that the pull-up resistor.

7、Single-bus communication（ONE-WIRE）

7.1 Typical circuits for single bus

Microprocessor and AM2301 connection typical application circuit is shown in Figure 4. Single bus communication mode, pull the SDA microprocessor I / O port is connected.

Special instructions of the single-bus communication:

1. Typical application circuit recommended in the short cable length of 30 meters on the 5.1K pull-up resistor pullup resistor according to the actual situation of lower than 30 m.
2. With 3.3V supply voltage, cable length shall not be greater than 100cm. Otherwise, the line voltage drop will lead to the sensor power supply, resulting in measurement error.
3. Read the sensor minimum time interval for the 2S; read interval is less than 2S, may cause the temperature and humidity are not allowed or communication is unsuccessful, etc..
4. Temperature and humidity values are each read out the results of the last measurement For real-time data that need continuous read twice, we recommend repeatedly to read sensors, and each read sensor interval is greater than 2 seconds to obtain accuratethe data.

Anexos 2: HOJA DE DATOS TÉCNICOS DEL SENSOR MQ135

TECHNICAL DATA MQ-135 GAS SENSOR

FEATURES

Wide detecting scope — Fast response and High sensitivity
Stable and long life — Simple drive circuit

APPLICATION

They are used in air quality control equipments for buildings/offices, are suitable for detecting of NH3,NOx, alcohol, Benzene, smoke,CO_2 ,etc.

SPECIFICATIONS

A. Standard work condition

Symbol	Parameter name	Technical condition	Remarks
Vc	Circuit voltage	5V±0.1	AC OR DC
V_H	Heating voltage	5V±0.1	ACOR DC
R_L	Load resistance	can adjust	
R_H	Heater resistance	33Ω±5%	Room Tem
P_H	Heating consumption	less than 800mw	

B. Environment condition

Symbol	Parameter name	Technical condition	Remarks
Tao	Using Tem	-10 -45	
Tas	Storage Tem	-20 -70	
R_H	Related humidity	less than 95%Rh	
O_2	Oxygen concentration	21%(standard condition)Oxygen concentration can affect sensitivity	minimum value is over 2%

C. Sensitivity characteristic

Symbol	Parameter name	Technical parameter	Remark 2
Rs	Sensing Resistance	30KΩ-200KΩ (100ppm NH_3)	Detecting concentration scope 10ppm-300ppm NH_3 10ppm-1000ppm Benzene 10ppm-300ppm Alcohol
α (200/50) NH_3	Concentration Slope rate	≤0.65	
Standard Detecting Condition	Temp: 20 ±2 Vc:5V±0.1 Humidity: 65%±5% Vh: 5V±0.1		
Preheat time	Over 24 hour		

D. Structure and configuration, basic measuring circuit

	Parts	Materials
1	Gas sensing layer	SnO_2
2	Electrode	Au
3	Electrode line	Pt
4	Heater coil	Ni-Cr alloy
5	Tubular ceramic	Al_2O_3
6	Anti-explosion network	Stainless steel gauze (SUS316 100-mesh)
7	Clamp ring	Copper plating Ni
8	Resin base	Bakelite
9	Tube Pin	Copper plating Ni

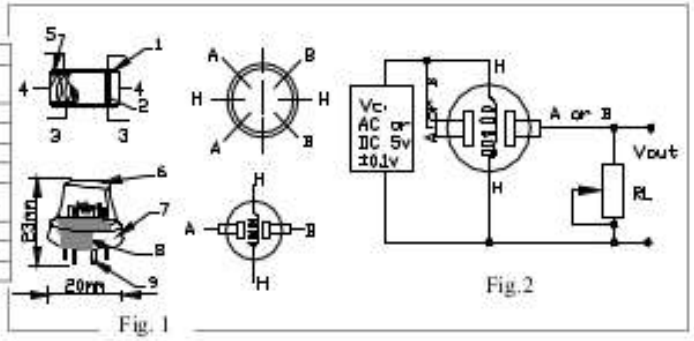
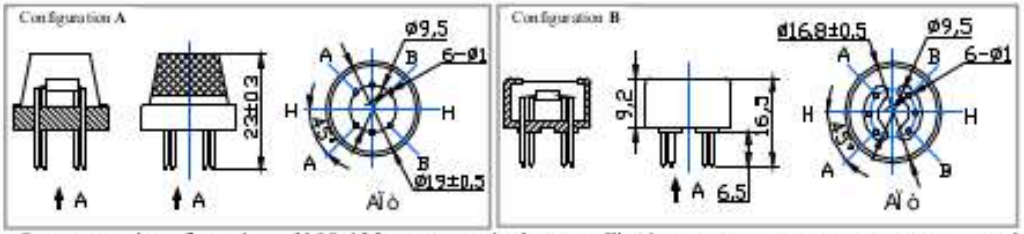


Fig.2

Fig. 1



Structure and configuration of MQ-135 gas sensor is shown as Fig. 1 (Configuration **A or B**), sensor composed by micro AL_2O_3 ceramic tube, Tin Dioxide (SnO_2) sensitive layer, measuring electrode and heater are fixed into a crust made by plastic and stainless steel net. The heater provides necessary work conditions for work of sensitive

components. The enveloped MQ-135 have 6 pin ,4 of them are used to fetch signals, and other 2 are used for providing heating current.

Electric parameter measurement circuit is shown as Fig.2

E. Sensitivity characteristic curve

Fig.2 sensitivity characteristics of the MQ-135

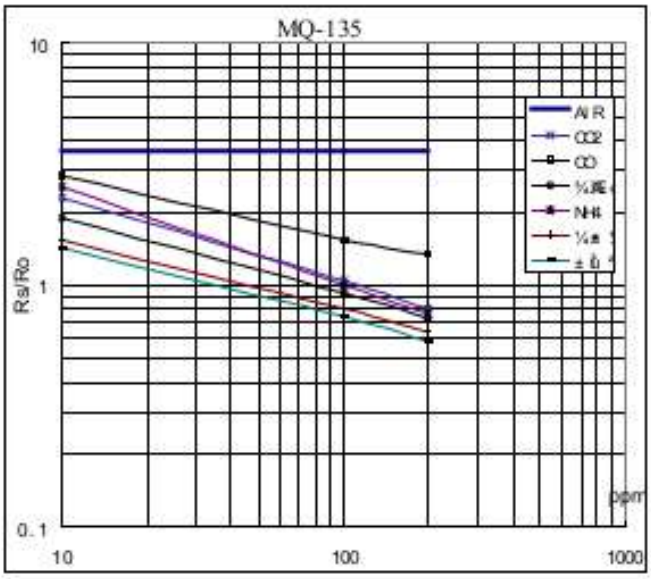


Fig.3 is shows the typical sensitivity characteristics of the MQ-135 for several gases.
in their: Temp: 20
Humidity: 65%
O_2 concentration 21%
RL=20kΩ
Ro: sensor resistance at 100ppm of NH_3 in the clean air.
Rs:sensor resistance at various concentrations of gases.

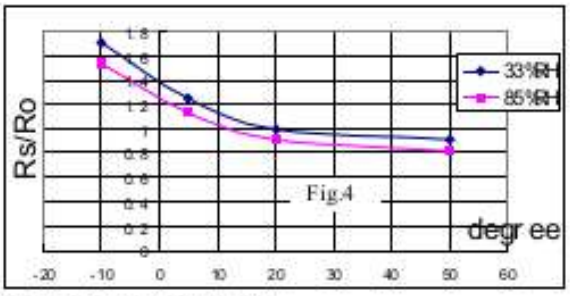
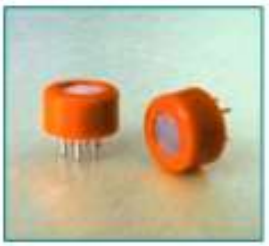


Fig.4 is shows the typical dependence of the MQ-135 on temperature and humidity.
Ro: sensor resistance at 100ppm of NH_3 in air at 33%RH and 20 degree.
Rs: sensor resistance at 100ppm of NH_3 at different temperatures and humidities.

SENSITVITY ADJUSTMENT

Resistance value of MQ-135 is difference to various kinds and various concentration gases. So, When using this components, sensitivity adjustment is very necessary. we recommend that you calibrate the detector for 100ppm NH_3 or 50ppm Alcohol concentration in air and use value of Load resistancethat(R_L) about 20 KΩ(10KΩ to 47 KΩ).

When accurately measuring, the proper alarm point for the gas detector should be determined after considering the temperature and humidity influence.



Anexos 3: HOJA DE DATOS TÉCNICOS DEL SENSOR HX711

24-Bit Analog-to-Digital Converter (ADC) for Weigh Scales

DESCRIPTION

Based on Avia Semiconductor's patented technology, HX711 is a precision 24-bit analog-to-digital converter (ADC) designed for weigh scales and industrial control applications to interface directly with a bridge sensor.

The input multiplexer selects either Channel A or B differential input to the low-noise programmable gain amplifier (PGA). Channel A can be programmed with a gain of 128 or 64, corresponding to a full-scale differential input voltage of ±20mV or ±40mV respectively, when a 5V supply is connected to AVDD analog power supply pin. Channel B has a fixed gain of 32. On-chip power supply regulator eliminates the need for an external supply regulator to provide analog power for the ADC and the sensor. Clock input is flexible. It can be from an external clock source, a crystal, or the on-chip oscillator that does not require any external component. On-chip power-on-reset circuitry simplifies digital interface initialization.

There is no programming needed for the internal registers. All controls to the HX711 are through the pins.

FEATURES

- **Two selectable differential input channels**
- **On-chip active low noise PGA with selectable gain of 32, 64 and 128**
- **On-chip power supply regulator for load-cell and ADC analog power supply**
- **On-chip oscillator requiring no external component with optional external crystal**
- **On-chip power-on-reset**
- **Simple digital control and serial interface: pin-driven controls, no programming needed**
- **Selectable 10SPS or 80SPS output data rate**
- **Simultaneous 50 and 60Hz supply rejection**
- **Current consumption including on-chip analog power supply regulator: normal operation < 1.5mA, power down < 1uA**
- **Operation supply voltage range: 2.6 ~ 5.5V**
- **Operation temperature range: -40 ~ +85℃**
- **16 pin SOP-16 package**

APPLICATIONS

- **Weigh Scales**
- **Industrial Process Control**

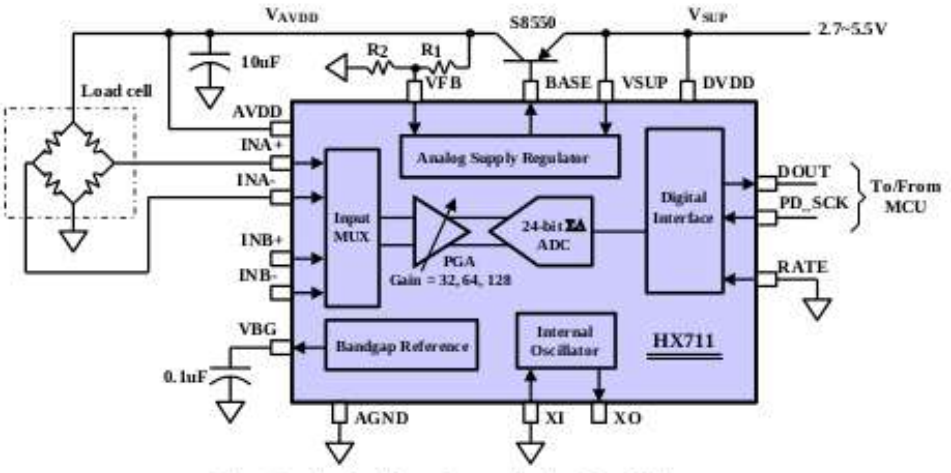
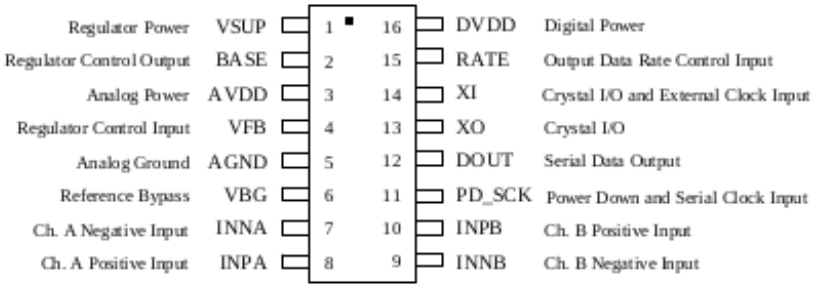


Fig. 1 Typical weigh scale application block diagram

Pin Description



SOP-16L Package

Pin #	Name	Function	Description
1	VSUP	Power	Regulator supply: 2.7 ~ 5.5V
2	BASE	Analog Output	Regulator control output（NC when not used）
3	AVDD	Power	Analog supply: 2.6 ~ 5.5V
4	VFB	Analog Input	Regulator control input（connect to AGND when not used）
5	AGND	Ground	Analog Ground
6	VBG	Analog Output	Reference bypass output
7	INA-	Analog Input	Channel A negative input
8	INA+	Analog Input	Channel A positive input
9	INB-	Analog Input	Channel B negative input
10	INB+	Analog Input	Channel B positive input
11	PD_SCK	Digital Input	Power down control (high active) and serial clock input
12	DOUT	Digital Output	Serial data output
13	XO	Digital I/O	Crystal I/O (NC when not used）
14	XI	Digital Input	Crystal I/O or external clock input, 0: use on-chip oscillator
15	RATE	Digital Input	Output data rate control, 0: 10Hz; 1: 80Hz
16	DVDD	Power	Digital supply: 2.6 ~ 5.5V

Table 1 Pin Description

KEY ELECTRICAL CHARACTERISTICS

Parameter	Notes	MIN	TYP	MAX	UNIT
Full scale differential input range	V(inp)-V(inn)		±0.5(AVDD/GAIN)		V
Common mode input		AGND+1.2		AVDD-1.3	V
Output data rate	Internal Oscillator, RATE = 0		10		Hz
	Internal Oscillator, RATE = DVDD		80		
	Crystal or external clock, RATE = 0		f_{clk}/1,105,920		
	Crystal or external clock, RATE = DVDD		f_{clk}/138,240		
Output data coding	2's complement	800000		7FFFFF	HEX
Output settling time (1)	RATE = 0		400		ms
	RATE = DVDD		50		
Input offset drift	Gain = 128		0.2		mV
	Gain = 64		0.4		
Input noise	Gain = 128， RATE = 0		50		nV(rms)
	Gain = 128， RATE = DVDD		90		
Temperature drift	Input offset（Gain = 128）		±6		nV/℃
	Gain（Gain = 128）		±5		ppm/℃
Input common mode rejection	Gain = 128， RATE = 0		100		dB
Power supply rejection	Gain = 128， RATE = 0		100		dB
Reference bypass（V_{BG}）			1.25		V
Crystal or external clock frequency		1	11.0592	20	MHz
Power supply voltage	DVDD	2.6		5.5	V
	AVDD，VSUP	2.6		5.5	
Analog supply current (including regulator)	Normal		1400		μA
	Power down		0.3		
Digital supply current	Normal		100		μA
	Power down		0.2		

（1）Settling time refers to the time from power up, reset, input channel change and gain change to valid stable output data.

Table 2 Key Electrical Characteristics

Anexos 4: HOJA DE DATOS TÉCNICOS DEL ARDUINO MEGA 2560

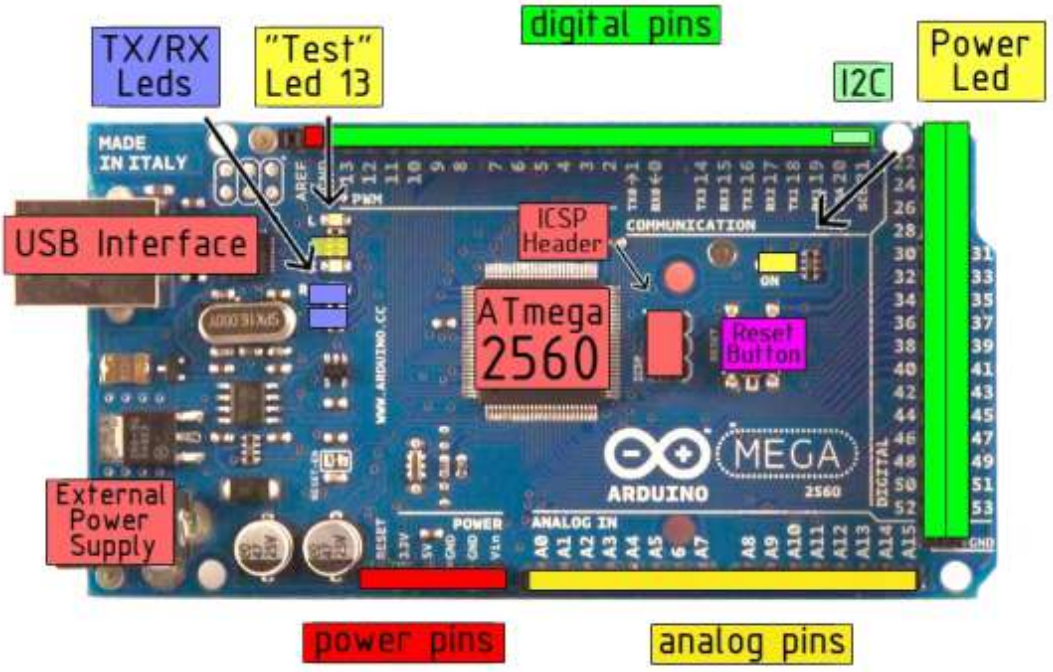
Technical Specification

EAGLE files: arduino-mega2560-reference-design.zip Schematic: arduino-mega2560-schematic.pdf

Summary

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	54 (of which 14 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz

the board



Power

The Arduino Mega2560 can be powered via the USB connection or with an external power supply. The power source is selected automatically. External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The Mega2560 differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega8U2 programmed as a USB-to-serial converter.

The power pins are as follows:

- **VIN.** The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- **5V.** The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply.
- **3V3.** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- **GND.** Ground pins.

Memory

The ATmega2560 has 256 KB of flash memory for storing code (of which 8 KB is used for the bootloader), 8 KB of SRAM and 4 KB of EEPROM (which can be read and written with the EEPROM library).

Input and Output

Each of the 54 digital pins on the Mega can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

- **Serial: 0 (RX) and 1 (TX); Serial 1: 19 (RX) and 18 (TX); Serial 2: 17 (RX) and 16 (TX); Serial 3: 15 (RX) and 14 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. Pins 0 and 1 are also connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip .
- **External Interrupts: 2 (interrupt 0), 3 (interrupt 1), 18 (interrupt 5), 19 (interrupt 4), 20 (interrupt 3), and 21 (interrupt 2).** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
- **PWM: 0 to 13.** Provide 8-bit PWM output with the analogWrite() function.
- **SPI: 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS).** These pins support SPI communication, which, although provided by the underlying hardware, is not currently included in the Arduino language. The SPI pins are also broken out on the ICSP header, which is physically compatible with the Duemilanove and Diecimila.
- **LED: 13.** There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.
- **I^2C: 20 (SDA) and 21 (SCL).** Support I^2C (TWI) communication using the Wire library (documentation on the Wiring website). Note that these pins are not in the same location as the I^2C pins on the Duemilanove.

The Mega2560 has 16 analog inputs, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and analogReference() function.

Communication

The Arduino Mega2560 has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega2560 provides four hardware UARTs for TTL (5V) serial communication. An ATmega8U2 on the board channels one of these over USB and provides a virtual com port to software on the computer (Windows machines will need a .inf file, but OSX and Linux machines will recognize the board as a COM port automatically. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the ATmega8U2 chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A SoftwareSerial library allows for serial communication on any of the Mega's digital pins.

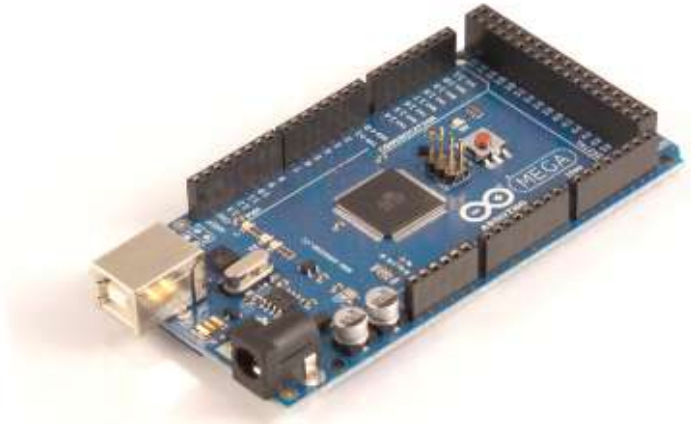
The ATmega2560 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation on the Wiring website for details. To use the SPI communication, please see the ATmega2560 datasheet.

Programming

The Arduino Mega2560 can be programmed with the Arduino software (download). For details, see the reference and tutorials.

The Atmega2560 on the Arduino Mega comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files).

You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see these instructions for details.



Anexos 5: HOJA DE DATOS TÉCNICOS DEL MODULO ESP-01

1. Preambles

ESP-01 WiFi module is developed by Ai-thinker Team. core processor ESP8266 in smaller sizes of the module encapsulates Tensilica L106 integrates industry-leading ultra low power 32-bit MCU micro, with the 16-bit short mode, Clock speed support 80 MHz, 160 MHz, supports the RTOS, integrated Wi-Fi MAC/BB/RF/PA/LNA, on-board antenna.

The module supports standard IEEE802.11 b/g/n agreement, complete TCP/IP protocol stack. Users can use the add modules to an existing device networking, or building a separate network controller.

ESP8266 is high integration wireless SOCs, designed for space and power constrained mobile platform designers. It provides unsurpassed ability to embed Wi-Fi capabilities within other systems, or to function as a standalone application, with the lowest cost, and minimal space requirement.

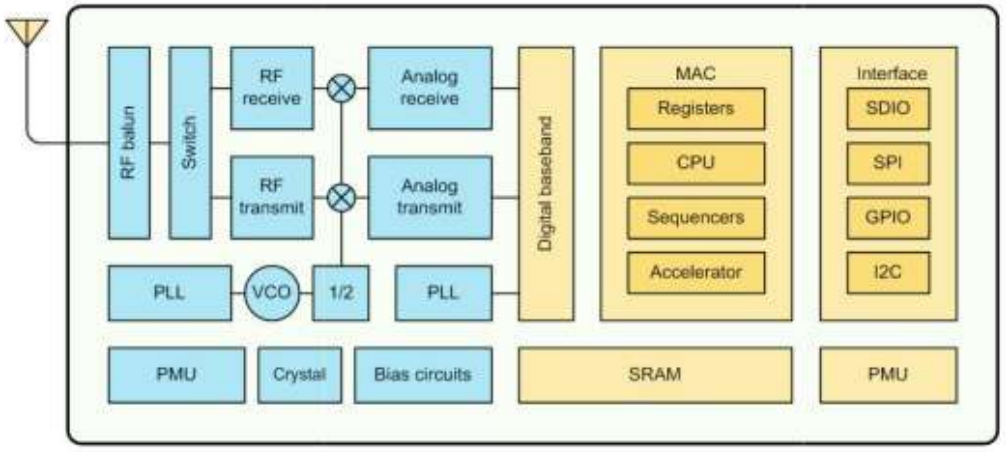


Figure 1 ESP8266EX Block Diagram

ESP8266EX offers a complete and self-contained Wi-Fi networking solution; it can be used to host the application or to offload Wi-Fi networking functions from another application processor.

When ESP8266EX hosts the application, it boots up directly from an external flash. In has integrated cache to improve the performance of the system in such applications.

Alternately, serving as a Wi-Fi adapter, wireless internet access can be added to any micro controllerbased design with simple connectivity (SPI/SDIO or I2C/UART interface).

ESP8266EX is among the most integrated WiFi chip in the industry; it integrates the antenna switches, RF balun, power amplifier, low noise receive amplifier, filters, power management modules, it requires minimal external circuitry, and the entire solution, including front-end module, is designed to occupy minimal PCB area.

ESP8266EX also integrates an enhanced version of Tensilica's L106 Diamond series 32-bit processor, with on-chip SRAM, besides the Wi-Fi functionalities. ESP8266EX is often integrated with external sensors and other application specific devices through its GPIOs; codes for such applications are provided in examples in the SDK.

Espressif Systems' Smart Connectivity Platform (ESCP) demonstrates sophisticated system-level features include fast sleep/wake context switching for energy-efficient VoIP, adaptive radio biasing. for low-power operation, advance signal processing, and spur cancellation and radio co-existence features for common cellular, Bluetooth, DDR, LVDS, LCD interference mitigation.

1.1. Features

- 802.11 b/g/n
- Integrated low power 32-bit MCU
- Integrated 10-bit ADC
- Integrated TCP/IP protocol stack
- Integrated TR switch, balun, LNA, power amplifier and matching network
- Integrated PLL, regulators, and power management units
- Supports antenna diversity
- Wi-Fi 2.4 GHz, support WPA/WPA2
- Support STA/AP/STA+AP operation modes
- Support Smart Link Function for both Android and iOS devices
- Support Smart Link Function for both Android and iOS devices
- SDIO 2.0, (H) SPI, UART, I2C, I2S, IRDA, PWM, GPIO

1.2. Parameters

Table 1 below describes the major parameters.

Table 1 Parameters

Categories	Items	Values
WiFi Paramters	WiFi Protocles	802.11 b/g/n
	Frequency Range	2.4GHz-2.5GHz (2400M-2483.5M)
Hardware Paramaters	Peripheral Bus	UART/HSPI/I2C/I2S/Ir Remote Contorl
		GPIO/PWM
	Operating Voltage	3.0~3.6V
	Operating Current	Average value: 80mA
	Operating Temperature Range	-40°~125°
	Ambient Temperature Range	Normal temperature
	Package Size	14.3mm*24.8mm*3mm
	External Interface	N/A
Software Parameters	Wi-Fi mode	station/softAP/SoftAP+station
	Security	WPA/WPA2
	Encryption	WEP/TKIP/AES
	Firmware Upgrade	UART Download / OTA (via network) / download and write firmware via host
	Ssoftware Development	Supports Cloud Server Development / SDK for custom firmware development
	Network Protocols	IPv4, TCP/UDP/HTTP/FTP
	User Configuration	AT Instruction Set, Cloud Server, Android/iOS App

2. Pin Descriptions

There are altogether 8 pin counts, the definitions of which are described in Table 2 below.

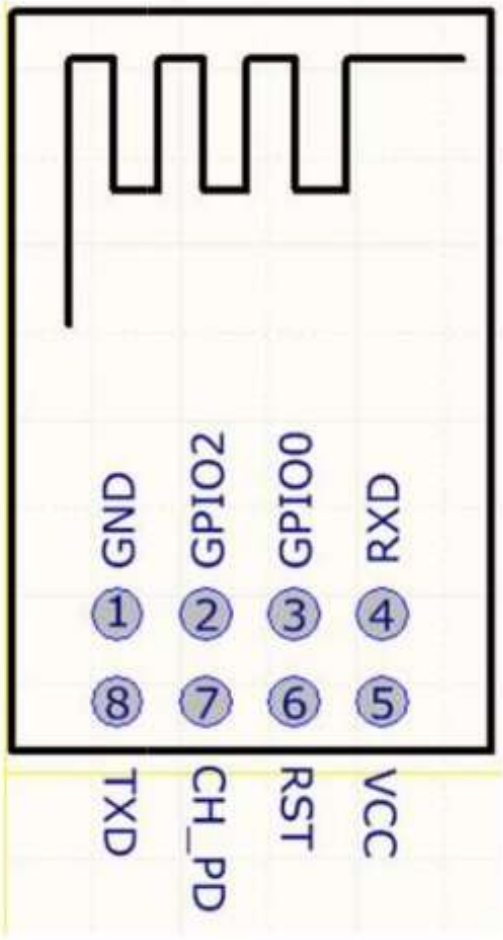


Table 2 ESP-01 Pin design

Table 2 Pin Descriptions

NO.	Pin Name	Function
1	GND	GND
2	GPIO2	GPIO,Internal Pull-up
3	GPIO0	GPIO,Internal Pull-up
4	RXD	UART0,data received pin RXD
5	VCC	3.3V power supply (VDD)
6	RST	1) External reset pin, active low 2) Can loft or external MCU ,
7	CH_PD	Chip enable pin. Active high
8	TXD	UART0,data send pin RXD

Anexos 6: CÓDIGO FUENTE PARA ENVIAR LOS DATOS AL SERVIDOR LOCAL MEDIANTE ARDUINO MEGA Y EL MODULO ESP-01

```
#include "SoftwareSerial.h" //Librer a comunicaci n serie
#include "HX711.h" //librer a del sensor HX711x
#include <DHT.h> //librer a del sensor DHT21
#include <CapacitiveSensor.h> //librer a del sensor capacitivo
SoftwareSerial softSerial(2, 3); // RX, TX
const int baudRate = 9600;
char* SSDI = "WiFi"; // Nombre de la red wifi
char* password = "Password";// contrase a de la red
//nombre de dominio y ruta URL o direcci n IP
const char* serverName = "http://localhost/arduino/enviar_datos.php";
//Factor de calibraci n HX711
#define calibration_factor -1770.0
#define LOADCELL_DOUT_PIN  5 // Datos del HX711
#define LOADCELL_SCK_PIN  4 //Serial Clock HX711
HX711 scale;
int s_analogica_mq135=0; //Pin analogico A0
// Definimos el pin digital donde se conecta el sensor
#define DHTPIN 8
// Dependiendo del tipo de sensor
#define DHTTYPE DHT21
// Inicializamos el sensor DHT21
DHT dht(DHTPIN, DHTTYPE);
//Definimos los pines de los sensores capacitivos
CapacitiveSensor   cs_30_32 = CapacitiveSensor(30,32);
CapacitiveSensor   cs_30_33 = CapacitiveSensor(30,33);
CapacitiveSensor   cs_30_34 = CapacitiveSensor(30,34);
CapacitiveSensor   cs_30_35 = CapacitiveSensor(30,35);

CapacitiveSensor   cs_30_36 = CapacitiveSensor(30,36);
CapacitiveSensor   cs_30_37 = CapacitiveSensor(30,37);
CapacitiveSensor   cs_30_38 = CapacitiveSensor(30,38);
CapacitiveSensor   cs_30_39 = CapacitiveSensor(30,39);

void setup()
{
```

```
Serial.begin(baudRate); //se inicia la comunicaci n serial
softSerial.begin(baudRate);
delay(1000);

//calibraci n autom tica del sensor capacitivo
cs_30_32.set_CS_AutocaL_Millis(0xFFFFFFF);

scale.begin(LOADCELL_DOUT_PIN, LOADCELL_SCK_PIN);
scale.set_scale(calibration_factor); //Factor de calibraci n
scale.tare();  //Suponiendo que no haya peso en la b scula al inicio,
             //restablezca la b scula a 0
Serial.println(); //Salto de linea
Serial.print("CONECT NDOSE A LA RED : ");
Serial.println(ssid);//imprime el nombre de la red
WiFi.begin(ssid, password);//establece la conexi n con la red
while (WiFi.status() != WL_CONNECTED){
delay(500);
Serial.println("CONECTANDOSE...");
//esparando la conexion
}
Serial.println("");
Serial.println("WiFi conectado"); //Imprime Red WiFi conectado
Serial.println("DIRECCION IP:");
Serial.println(WiFi.localIP()); //Imprime la direcci n IP

Serial.print("Sensores: ");
Serial.print("Peso: ");
Serial.print(scale.get_units(), 3); //unidad de escala
Serial.print("  lbs\t");
// Leemos la humedad relativa
float h = dht.readHumidity();
// Leemos la temperatura en grados cent grados
float t = dht.readTemperature();

// Comprobamos si ha habido alg n error en la lectura
if (isnan(h) || isnan(t)) {
Serial.println("Error-DHT21");
```

```
return;}

Serial.print("Temperatura: ");
Serial.print(t);
Serial.println(" *C ");
Serial.print("  Humedad: ");
Serial.print(h);
Serial.print(" %\t");

s_analogica_mq135 = analogRead(0);
Serial.print("   Co2: ");
Serial.print(s_analogica_mq135, DEC);
Serial.print("  ppm\t");


}

void loop()
{ //variables de medion de los sensores capacitivos
   long total9  =  cs_30_32.capacitiveSensor(30);
   long total10 =  cs_30_33.capacitiveSensor(30);
   long total11 =  cs_30_34.capacitiveSensor(30);
   long total12 =  cs_30_35.capacitiveSensor(30);
   long total13 =  cs_30_36.capacitiveSensor(30);
   long total14 =  cs_30_37.capacitiveSensor(30);
   long total15 =  cs_30_38.capacitiveSensor(30);
   long total16 =  cs_30_39.capacitiveSensor(30);

 //Contador de poblacion
 if(total9  >= 35500) { int t1=t1+1;}
 if(total10 >= 25600) { int t2=t2+1;}
 if(total11 >= 32600) { int t3=t3+1;}
 if(total12 >= 28900) { int t4=t4+1;}
 if(total13 >= 30600) { int t5=t5+1;}
 if(total14 >= 27000) { int t6=t6+1;}
 if(total15 >= 32120) { int t7=t7+1;}
 if(total16 >= 28500) { int t8=t8+1;}
```

```
int poblacion = t1+t2+t3+t4+t5+t6+t7+t8;

//Verifica el estado de la conexi n WiFi
if(WiFi.status()== WL_CONNECTED){
HTTPClient http;

// nombre del dominio con ruta URL o direcci n IP con ruta
http.begin(serverName);

// Prepare sus datos de solicitud HTTP POST
String httpRequestData = "Temperatura=" + t + "&Humedad=" + h
+ "&Co2=" + s_analogica_mq135 + "&Poblacion=" + String(poblacion)
+ "&Peso=" + String(scale.get_units()) "";
Serial.print("httpRequestData: ");
Serial.println(httpRequestData);
// enviar solicitud HTTP POST
int httpResponseCode = http.POST(httpRequestData);

if (httpResponseCode>0) {
Serial.print("HTTP Response code: ");
Serial.println(httpResponseCode);
}
else {
Serial.print("Error code: ");
Serial.println(httpResponseCode);
}
http.end();
}
else {
serial.println("WiFi Disconnected");
}
//Enviar una solicitud HTTP POST cada 30 segundos
delay(30000);
}
```

Anexos 7: CÓDIGO FUENTE PARA LA CONEXIÓN CON LA BASE DE DATOS

```
<?php
  // Parametros a configurar para la conexion de la base de datos
  $host = "localhost";    // direccion de la BD
  $basededatos = "BeeApp";    // nombre de la BD
  $usuariodb = "phpmyadmin";    // usuario de nuestra BD
  $clavedb = "debian";    // clave de la BD

  //Lista de Tablas
  $tabla_db1 = "DatosPoblacionPeso";     // Poblacion, Peso
    $tabla_db2 = "DatosSensores";    //Temperatura, Humedad, Co2

  $conexion = new mysqli($host,$usuariodb,$clavedb,$basededatos);

  if ($conexion->connect_errno) {
      echo "Nuestro sitio experimenta fallos....";
      exit();
  }
?>
```

Anexos 8: CÓDIGO FUENTE PARA ENVIAR LA INFORMACIÓN A LA BASE DE DATOS

```
<?php

include_once( $_SERVER['DOCUMENT_ROOT'] . "/arduino/config.inc.php" );
include_once(DIR_INC . "class.mysql.inc.php");
$bd = new class_mysql();
$bd->insertar($_GET['Temperatura'] , $_GET['Humedad'], $_GET['Co2'],
$_GET['Poblacion'], $_GET['Peso']);
$resultado=$bd->listar();

?>
```

Anexos 9: MANUAL BÁSICO PARA LA APICULTURA

EQUIPO DE PROTECCIÓN Y MANEJO

En la apicultura moderna el equipo de protección del apicultor es importante, sobre todo cuando se trata de trabajar con abejas africanizadas, las cuales son muy defensivas. Las abejas defienden sus colonias y pueden picar a las personas que las manejan. Para evitar esto los apicultores usan ropa especial. Estas son las partes del equipo de protección del apicultor [59]:

- El **velo** sirve para proteger la cabeza y la cara del apicultor. Consta de una careta de malla mosquitero negra que permite ver contra el reflejo del sol y el resto del es una pieza que puede ser de diferentes materiales desde una trama de hilo cáñamo hasta manta, en la parte inferior de esta tiene una jareta que permite pegarlo al cuerpo.
- El **overol** es un vestido de una sola pieza, es decir, que el pantalón y la camisa van unidos. A las abejas les molesta la ropa de color negro, rojo o verde oscuro. En cambio la ropa de color blanco no les molesta. La ropa debe ser de algodón porque no molesta a las abejas tanto como la de lana o la de cuero. Los olores de los animales, que quedan en la lana y el cuero irritan a las abejas.
- Los **guantes** tienen que ser de cuero liso y suave. Estos sirven para protegerse las manos. Igual que otros equipos, los guantes deben lavarse cada vez que están sucios y guardarlos secos, colgados o bien doblarlos. Las abejas pueden picar los pies.
- Para evitarlo se usan las **botas** o los **zapatos altos**. Esto le ayuda a protegerse los pies. No es recomendable usar guaraches cuando va a trabajar con las abejas [59].

EL EQUIPO DE MANEJO

Es muy importante tener las herramientas necesarias para trabajar las colmenas.

- EL **ahumador**: para el manejo de una colmena, esta herramienta es absolutamente necesaria. Produce humo con la finalidad de controlar a las abejas,

haciéndolas huir de las partes de la colmena que se quiere examinar.

- **Espátula o cuña**: consiste en una pieza de acero afilada por un extremo para separar todas las partes de la colmena que están adheridas con propóleos. El otro extremo de la cuña tiene una forma redonda y sirve para raspar la cera que se encuentra adherida en las paredes de la colmena. Cuando se está trabajando con las colmenas, ésta herramienta se debe tener todo el tiempo a la mano [59].

INSTALACIÓN DE UN APIARIO

Al colocar nuestros apiarios (también conocido como colmenar) , los principales factores a tener en cuenta son:

- Ubicar el apiario cerca de donde exista abundancia de flores, ya que de ellas depende la producción de miel y polen.
- La colmena se orientará de manera que el sol dé en la piquera cuanto antes, porque ello incentivará a las abejas a empezar a trabajar más temprano.
- Evitar lugares húmedos, y si es una región de mucho calor, ubicar las colmenas en áreas sombreadas, pero sin ser sombra cerrada.
- El lugar donde se coloquen las colmenas debe estar limpio para evitar que se alojen hormigas u otros enemigos de las abejas. La colmena se coloca sobre una base resistente que tenga una altura mínima de 20 cm del suelo.
- El apiario debe situarse en un lugar nivelado y seco, donde se pueda transitar libremente por detrás de las colmenas para realizar las diferentes actividades de manejo.
- Proteger el apiario de vientos fríos y fuertes con la instalación de arbustos o barreras naturales que formen cercas vivas [59].

PASOS PARA LA REVISIÓN

La cuña y ahumador en mano la persona que revisará la colmena se ubicará en la parte lateral de la misma. Nunca se debe colocar enfrente de la entrada en la piquera porque dificultará la entrada y salida de abejas, provocando su defensibilidad, resultando

difícil la revisión de la colmena.

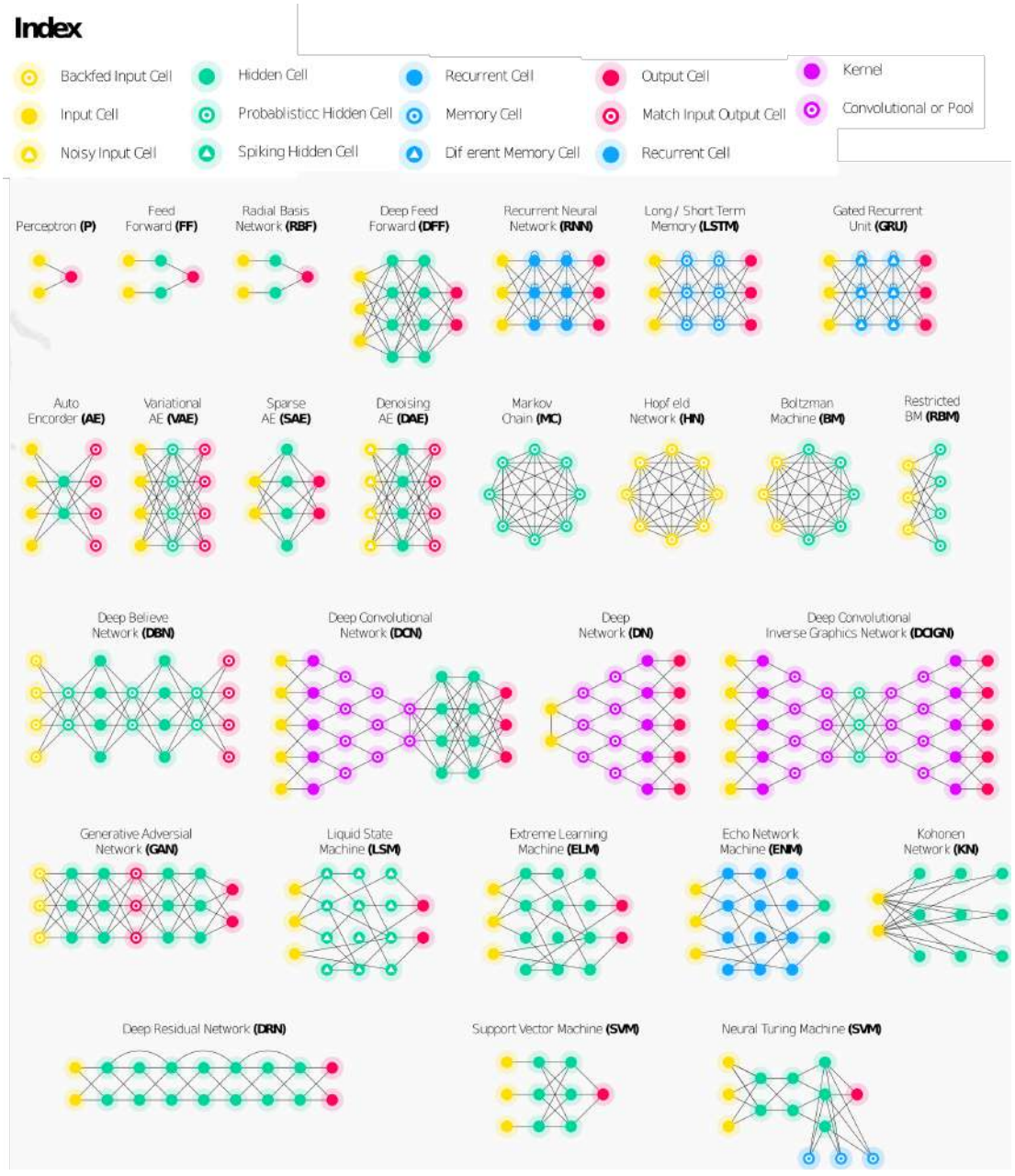
1. Echar tres o cuatro veces humo a la piquera de la colmena que vamos a revisar así como a las colmenas que se encuentran a los lados de ésta para que las abejas no se alteren demasiado.
2. Quitar el techo y colocarlo en el suelo y hacia arriba para colocar las alzas en forma esquinada, en caso de que la colmena tenga alzas.
3. Levantar con cuidado la tapa con ayuda de la cuña, echando humo por el hueco que se haya abierto y seguir echando humo hasta levantar la tapa por completo, la que se colocará a un lado de la colmena en el suelo y hacia arriba.
4. De igual forma y en caso de que la colmena tenga alzas, se irán quitando una a una y se colocarán sobre el techo que previamente hemos puesto en el suelo. Acto seguido se procederá a sacar y revisar uno por uno los bastidores de la cámara de cría, echando humo por encima de los bastidores cada vez que sea necesario calmar a las abejas.
5. De ser necesario por si las abejas están bravas y para evitar ataques, se cubrirán las alzas con la tapa.
6. Proceda en la revisión básica a satisfacer las necesidades que tenga la colmena, como curación, cambio de bastidores viejos, sustitución de reina, etc.
7. Ya revisada la cámara de cría y acomodados los bastidores tal y como estaban en un principio, colocamos las alzas con cuidado para no aplastar abejas. Para ello siempre echamos humo.
8. Se sacuden las abejas que se encuentren en la tapa y se coloca en la colmena.
9. Por último sacudimos en la piquera las abejas que se encuentren en el techo y lo colocamos.
10. Al terminar de revisar cada colmena es bueno escribir los detalles en la ficha de registro [59].

(a) Pasos para la revisión de la colmena

(b) Cambio de cámara de cría por el prototipo

Figura 4.1: Cambio de la colmena al prototipo del sistema de monitorea apícola.

Anexos 10: REDES NEURONALES, HOJA DE TRUCOS BÁSICA



Anexos 11: CÓDIGO FUENTE PARA LA PREDICCIÓN DE SERIES TEMPORALES

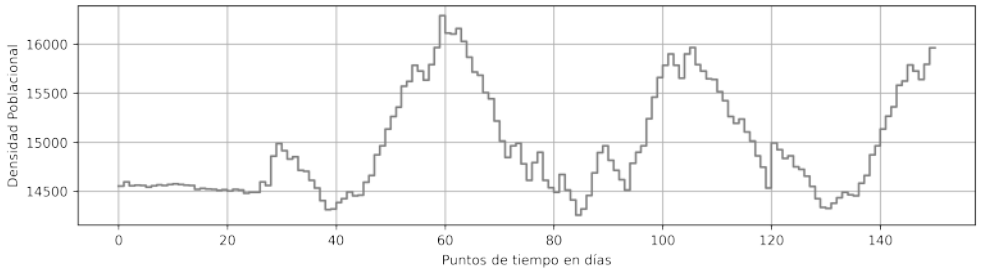
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import math
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Activation
from keras.layers import Dropout
from keras.layers import LSTM
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_squared_error

%matplotlib inline
```

```
# cargamos el conjunto de datos
df = pd.read_csv('DataPoblacion.csv', usecols=[1],␣
 ↪engine='python', skipfooter=0)
data = df.values
data = data.astype('float32')
print('Número de puntos de datos:',len(data))
```

```
Número de puntos de datos: 151
```

```
#Visualizamos el conjunto de datos
plt.figure(figsize=(12,3), dpi=300)
plt.step(df.index, df.Poblacion, color='tab:gray')
plt.grid()
plt.gca().set(xlabel="Puntos de tiempo en días",␣
 ↪ylabel="Densidad Poblacional")
plt.show()
```




```
# Conjunto de datos de entrenamiento y validación.
train_size = int(len(data) * 0.80) #Entrenamiento
test_size = len(data) - train_size #Validación
train, test = data[0:train_size,:], data[train_size:
 ↪len(data),:]
print('Entrenamiento:', len(train),'puntos de datos')
print('Validacion:', len(test),'puntos de datos')
```

```
Entrenamiento: 120 puntos de datos
Validacion: 31 puntos de datos
```

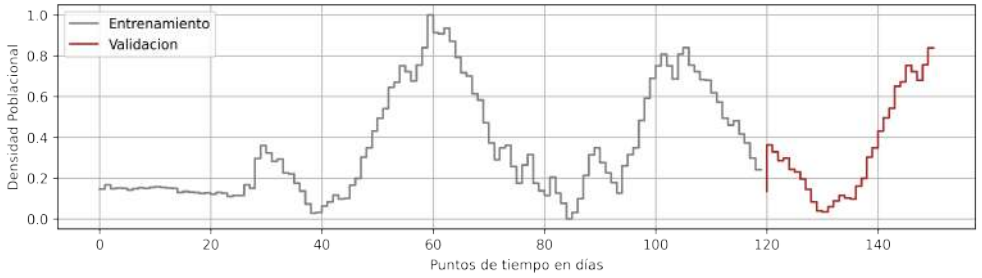
```
#escala el conjunto de datos
scaler = MinMaxScaler(feature_range=(0, 1))
data = scaler.fit_transform(data)
```

```
print('Min: %f, Max: %f' % (scaler.data_min_, scaler.
 ↪data_max_))
```

```
Min: 14256.000000, Max: 16289.000000
```

```
#normalizar el conjunto de datos
norm_train = scaler.transform(train)
norm_test = scaler.transform(test)
```

```
plt.figure(figsize=[12,3], dpi=300)
plt.xlabel('Puntos de tiempo en días')
plt.ylabel('Densidad Poblacional')
plt.grid()
plt.step(range(len(train)),norm_train,'gray')
plt.
 ↪step(range(len(train),len(train)+len(test)),norm_test
,'brown')
plt.legend(['Entrenamiento','Validacion']);
```




```
# convierte una matriz de valores en
# una matriz de conjunto de datos
def split_X_y(data, look_back = 1):
        dataX, dataY = [], []
        for i in range(len(data)-look_back-1):
                a = data[i:(i+look_back), 0]
                dataX.append(a)
                dataY.append(data[i + look_back, 0])
        return (np.array(dataX), np.array(dataY))
```

```
# remodela la entrada de los datos para
#los tiempos t-i ... t-2 t-1
look_back = 1 # punto de datos anterior para
#predecir el siguiente
trainX, trainY = split_X_y(norm_train, look_back)
testX, testY = split_X_y(norm_test, look_back)
```

```
#remodelar la entrada para que
# sea [muestras, pasos de tiempo, características]
trainX = np.reshape(trainX, (trainX.shape[0], 1, trainX.
 ↪shape[1]))
testX = np.reshape(testX, (testX.shape[0], 1, testX.
 ↪shape[1]))
```

```
# Creamos la Red Neuronal Artificial LSTM
model = Sequential()
model.add(LSTM(25, return_sequences=True,␣
 ↪input_shape=(trainX.shape[1],1), activation='tanh'))
model.add(Dropout(.2))
```

```
model.add(LSTM(25, return_sequences=False))
model.add(Dense(1))
model.compile(loss='mean_squared_error',␣
 ↪optimizer='adam')
history=model.fit(trainX, trainY, epochs=120,␣
 ↪batch_size=1, validation_data=(trainX, trainY),␣
 ↪verbose=2)
```

```
Train on 118 samples, validate on 118 samples
Epoch 1/120
 - 1s - loss: 0.1049 - val_loss: 0.0482
```

```
print(model.summary())#imprime el modelo de red
```

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              ␣
 ↪Param #
=================================================================
lstm_3 (LSTM)                (None, 1, 25)             2700
_________________________________________________________________
dropout_2 (Dropout)          (None, 1, 25)             0
_________________________________________________________________
lstm_4 (LSTM)                (None, 25)                5100
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 26
=================================================================
Total params: 7,826
Trainable params: 7,826
Non-trainable params: 0
```

```
from keras.utils.vis_utils import plot_model
from keras.models import Sequential
plot_model(model, to_file='model_plot.png',␣
 ↪show_shapes=True, show_layer_names=True)
```

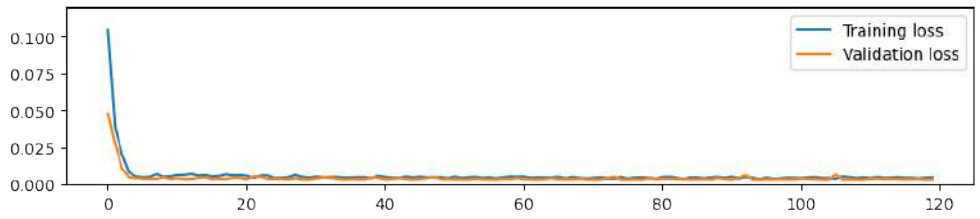
```
#Porcentaje de entrenamiento
from sklearn.metrics import r2_score
results=model.predict(trainX)
def adj_r2_score(r2, n, k):
    return 1-((1-r2)*((n-1)/(n-k-1)))
r2_test = (r2_score(results, trainY))*100
print("R-squared is: %f"%r2_test)
```

```
R-squared is: 93.905268
```

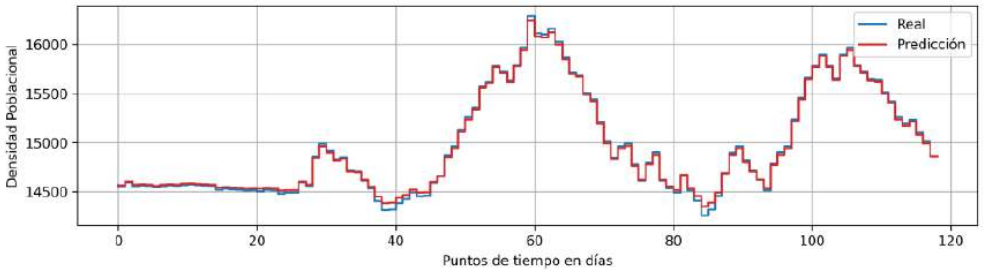
```
#Porcentaje de valdación
y_predic = model.predict(testX)
print('R-Squared: %f'%(r2_score(testY, y_predic)*100))
```

```
R-Squared: 92.451758
```

```
#Fluctuación del error durante el entrenamiento
plt.figure(figsize=(10,2), dpi=100)
plt.ylim(0.0, 0.12)
plt.plot(history.history['loss'], label='Training loss')
#plt.title('loss')
plt.plot(history.history['val_loss'], label='Validation␣
 ↪loss')
#plt.title('validate loss')
plt.legend();
plt.show()
```




```
plt.figure(figsize=[12,3], dpi=300)
plt.xlabel('Puntos de tiempo en días')
plt.ylabel('Densidad Poblacional')
plt.grid()
plt.step(range(len(train)-2),trainY,'gray')
plt.step(range(len(train)-2),y_pred,'brown')
plt.legend(['Real','Predicción']);
```




```
#Pronóstico de datos
df = pd.read_csv('DataPoblacionCSV.csv',␣
 ↪parse_dates=[0], header=None,index_col=0,␣
 ↪names=['Fecha','Poblacion'])

df['weekday']=[x.weekday() for x in df.index]
df['month']=[x.month for x in df.index]
df.head()
```

Fecha	Poblacion	weekday	month
2019-12-02	14553	0	12
2019-12-03	14551	1	12
2019-12-04	14595	2	12
2019-12-05	14556	3	12
2019-12-06	14562	4	12

```
ultimosDias = df['2020-04-01':'2020-04-30']
```

```
def series_to_supervised(data, n_in=1, n_out=1,␣
 ↪dropnan=True):
    n_vars = 1 if type(data) is list else data.shape[1]
    df = pd.DataFrame(data)
    cols, names = list(), list()
    # input sequence (t-n, ... t-1)
    for i in range(n_in, 0, -1):
        cols.append(df.shift(i))
        names += [('var%d(t-%d)' % (j+1, i)) for j in␣
 ↪range(n_vars)]
    # forecast sequence (t, t+1, ... t+n)
    for i in range(0, n_out):
        cols.append(df.shift(-i))
        if i == 0:
```

```
            names += [('var%d(t)' % (j+1)) for j in␣
 ↪range(n_vars)]
        else:
            names += [('var%d(t+%d)' % (j+1, i)) for j␣
 ↪in range(n_vars)]
    # put it all together
    agg = pd.concat(cols, axis=1)
    agg.columns = names
    # drop rows with NaN values
    if dropnan:
        agg.dropna(inplace=True)
    return agg
```

```
#Preparamos los datos para Test
scaledMerge=ultimosDias.drop('unidades',axis=1)
print(scaledMerge.values)

# marco como aprendizaje supervisado
reframed = series_to_supervised(scaledMerge, 30, 1)
newReframed=reframed.
 ↪drop(['var1(t)','var2(t)','var3(t)'],axis=1)
newReframed.head(7)
```

```
values = newReframed.values
testX = values[6:, :]
testY = testX.reshape((testX.shape[0], 1, testX.
 ↪shape[1]))
print(testX.shape)
print(testX)
ultDiaSemana = newReframed.index[len(newReframed.
 ↪index)-1].weekday()
```

```
def agregarNuevoValor(testX,nuevoValor,ultDiaSemana):
    for i in range(testX.shape[2]-3):
        testX[0][0][i] = testX[0][0][i+3]
    ultDiaSemana=ultDiaSemana+1
    if ultDiaSemana>6:
        ultDiaSemana=0
    testX[0][0][testX.shape[2]-3]=ultDiaSemana
    testX[0][0][testX.shape[2]-2]=12
    testX[0][0][testX.shape[2]-1]=nuevoValor
    return testX,ultDiaSemana
```

```
# Pronóstico para la mes de Abril
results=[]
for i in range(30):
    parcial=model.predict(x_test)
    results.append(parcial[0])
    print('pred',i,x_test)
    x_test,ultDiaSemana=agregarNuevoValor(x_test,
    parcial[0],ultDiaSemana)
```

```
#Visualizamos el pronóstico
prediccion1mesAbril = pd.DataFrame(inverted)
prediccion1mesAbril.columns = ['pronostico']
prediccion1mesAbril.plot()
#Guardamos los datos del pronóstico
prediccion1mesAbril.to_csv('pronostico.csv')
```

Anexos 12: CÓDIGO FUENTE PARA LA INTERFAZ WEB

```
#Librerias para machine learnin, deep learning
from flask import Flask, render_template, request,␣
 ↪url_for, redirect
from flask_mysqldb import MySQL
from werkzeug.utils import secure_filename

import os
import io

import pandas as pd
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.backends.backend_agg import␣
 ↪FigureCanvasAgg as FigureCanvas
from matplotlib.figure import Figure
import base64


import pygal
from pygal.style import Style
import json
import time

app = Flask(__name__)
app.config["data"] = "./info"

#conxion a la base de datos MySQL
app.config['MYSQL_HOST'] = 'localhost'
app.config['MYSQL_USER'] = 'phpmyadmin'
app.config['MYSQL_PASSWORD'] = 'debian'
app.config['MYSQL_DB'] = 'phpmyadmin'

mysql = MySQL(app)

#ruta de la interfaz web
@app.route('/', methods=['POST', 'GET'])
def Index():
    details = request.form
    cur = mysql.connection.cursor()
    cursor = mysql.connection.cursor()
    cur.execute('SELECT * FROM datos ORDER by ID DESC␣
 ↪LIMIT 1')
    cursor.execute('SELECT * FROM datos')
    tabla=cursor.fetchall()
    data = cur.fetchall()

    mysql.connection.commit()
    cur.close()

    img_url = 'static/images/bar_chart.svg?cache=' +␣
 ↪str(time.time())

    return render_template('index.html', enviar=data ,␣
 ↪table=tabla, image_url = img_url)

#ruta para insertar datos en la base MySQL
@app.route('/add', methods=['POST'])
def add():
    if request.method == 'POST':
        details = request.form
        bees = request.form['Bees']
        temp = request.form['Temp']
        hum = request.form['Hum']
```

```
        co2 = request.form['Co2']
        kg = request.form['Peso']
        cur = mysql.connection.cursor()
        #cur = mysql.get_db().cursor()
        cur.execute("INSERT INTO datos (Bees, Temp,␣
 ↪Hum, Co2, Peso) VALUES (%s,%s,%s,%s,%s)", (bees,␣
 ↪temp, hum, co2, kg))
        mysql.connection.commit()
        cur.close()
        return redirect(url_for('Index'))
    else:
        return 'Error'

#ruta para editar los datos de MySQL
@app.route('/edit/<id>', methods = ['POST', 'GET'])
def get_contact(id):
    cur = mysql.connection.cursor()
    cur.execute('SELECT * FROM datos WHERE id = %s',␣
 ↪(id))
    data = cur.fetchall()
    cur.close()
    print(data[0])
    return render_template('edit-datos.html', datos =␣
 ↪data[0])

#ruta para eliminar datos en MySQL
@app.route('/delete/<string:id>', methods =␣
 ↪['POST','GET'])
def delete_contact(id):
    cur = mysql.connection.cursor()
    cur.execute('DELETE FROM datos WHERE id = {0}'.
 ↪format(id))
    mysql.connection.commit()
    flash('Contact Removed Successfully')
    return redirect(url_for('Index'))

#ruta para mostrar las graficas de datos
@app.route("/bar")
def bar():
    with open('01.js','r') as bar_file:
        data = json.load(bar_file)
    custom_style =␣
 ↪Style(colors=('#991515','#1cbc7c'),background='transparent')

    chart = pygal.StackedLine(style = custom_style)
    mark_list = [x['mark'] for x in data]
    tourn_list = [x['tournament'] for x in data]
    chart.add('A',mark_list)
    chart.add('T',tourn_list)
    chart.x_labels = [x['year'] for x in data]
    chart.render_to_file('static/images/bar_chart.svg')
    img_url = 'static/images/bar_chart.svg?cache=' +␣
 ↪str(time.time())
    return render_template('a.html',image_url = img_url)

#ruta para seleccionar las variables a graficar
@app.route('/columnas', methods=['POST'])
def columnas():
    if request.method == 'POST':
        da = request.files['file']
        filename = secure_filename(da.filename)
```

```
        da.save(os.path.join(app.config["data"],␣
↪filename))
        df = pd.read_csv('./info/{}'.format(filename))
        columnas = {
            'columnas': df.columns.tolist(),
            'filename': filename
        }
        return render_template('generador.html',␣
↪columnas=columnas)

#ruta para mostrar los valores de prediccion
@app.route('/predecir',methods=["GET","POST"])
def predict():
        df= pd.read_csv("data/DataBase.csv")
        # Características y etiquetas
        df_X = df.text
        df_Y = df.label

    # Vectorización
        corpus = df_X
        cv = CountVectorizer()
        X = cv.fit_transform(corpus)

        naivebayes_model = open("models/model_lstm.
↪pkl","rb")
        clf = joblib.load(naivebayes_model)

        if request.method == 'POST':
                raw_text = request.form['rawtext']
                data = [raw_text]
                vect = cv.transform(data).toarray()
                my_prediction = clf.predict(vect)
                pred_score = clf.predict_proba(vect)
        return render_template('index.
↪html',prediction=my_prediction,pred_score=pred_score,)


#ruta para graficar los datos de entrada y salida
@app.route('/grafica', methods=['POST'])
def grafica():
    if request.method == 'POST':
        columna = request.form['columna']
        grafica = request.form['tipo']
        filename = request.form['filename']
        df = pd.read_csv('./info/{}'.
↪format(filename))[columna]

        plt.clf()
        if grafica == 'puntos':
            img = io.BytesIO()
            plt.title("la grafica por: "+columna)
            plt.figure(figsize=(10,4), dpi=100)
            plt.plot(df.head(100),'--')
            plt.savefig(img, format='png')
            img.seek(0)
            plot_url = base64.b64encode(img.getvalue()).
↪decode()
            return render_template('grafica.html',␣
↪imagen={ 'imagen': plot_url })
        elif grafica == 'lineas':
            print('lineas')
            img = io.BytesIO()
            plt.title("la grafica por: "+columna)
```

```
            plt.figure(figsize=(14,4), dpi=100)
            plt.plot(df.head(120))
            plt.savefig(img, format='png')
            img.seek(0)
            plot_url = base64.b64encode(img.getvalue()).
 ↪decode()
            return render_template('grafica.html',␣
 ↪imagen={ 'imagen': plot_url })
        else:
            img = io.BytesIO()

            datos = df.head(10).tolist()
            for i in range(len(datos)):
                plt.bar(i, datos[i], align = 'center')
            plt.savefig(img, format='png')
            img.seek(0)
            plot_url = base64.b64encode(img.getvalue()).
 ↪decode()
            return render_template('grafica.html',␣
 ↪imagen={ 'imagen': plot_url })


if __name__ == '__main__':
    app.run()
```