UNIVERSIDAD ESTATAL PENÍNSULA DE SANTA ELENA

FACULTAD DE SISTEMAS Y TELECOMUNICACIONES

CARRERA DE ELECTRÓNICA Y AUTOMATIZACIÓN

TRABAJO DE TITULACIÓN

Propuesta Tecnológica, previo a la obtención del Título de:

INGENIERA EN ELECTRÓNICA Y AUTOMATIZACIÓN

"IMPLEMENTACIÓN DE UNA ESTACIÓN DE MONITOREO Y ALARMA METEOROLÓGICO PORTÁTIL CON INTEGRACIÓN DE TECNOLOGÍA IoT"

AUTOR

LISBETTE PRISCILA RODRIGUEZ VILLACIS

PROFESOR TUTOR

ING. ÓSCAR W. GÓMEZ MORALES MGTR

LA LIBERTAD – ECUADOR 2024

AGRADECIMIENTO

Doy gracias a Dios por darme el valor para cumplir esta meta, a mi amada madre Magaly Villacis por su inagotable paciencia e inspiración a esforzarme cada día más, creyendo en mi a pesar de los obstáculos que se presentaron. Tu fortaleza ha sido el motivo de mi inspiración.

A mi padre el economista Ignacio Rodriguez quien en vida fue un buen profesional, la huella que dejaste en mi vida permitió que llegue a la culminación de esta meta.

A mi abuela Isabel Solís por darme su apoyo y guía desde siempre, por esos consejos que me ayudaron a formarme y nunca rendirme, este logro también es tuyo.

A mi Tío el Ingeniero Milton Villacis por apoyarme con sus consejos, conocimientos que me permitieron desenvolverme en toda mi etapa estudiantil.

A mis hermanos Darío, Jonathan y Arnold por el apoyo emocional, económico, e incentivación año tras año y su fe inquebrantable en mis capacidades.

A mi pareja Kevin Jiménez, por ser mi compañero incondicional y transmitirme seguridad en todo momento, lograr este objetivo es solo el comienzo de muchas metas que perseguiremos Juntos.

A mi tutor el Ingeniero Oscar Gómez por su ayuda, paciencia durante este proceso de titulación, guiándome con sus conocimientos para sacar adelante este proyecto.

DEDICATORIA

Dedico este proyecto a mi madre, quien me formo e inculco valores, tus sacrificios, amor incondicional y palabras de aliento han sido el impulso para superar cada obstáculo. Este logro es tanto tuyo como mío. A mi pareja a quien amo, en cada momento me brindaste alegría pese a la adversidad, tu compresión y ánimo nunca faltaron.

A mis más cercanos amigos que con su amistad y cada momento compartido ha sido esencial en este proyecto Jackson, Raúl, Bryan y al resto de personas que en su momento me guiaron. Su compañía ha hecho que este camino sea significativo y llevadero.

A todos ustedes, les agradezco por el apoyo.

APROBACIÓN DEL TUTOR

En mi calidad de Tutor/Tutora del trabajo de integración curricular denominado "**Implementación de una estación de monitoreo y alarma meteorológico portátil con integración de tecnología IoT**", elaborado por la estudiante Lisbette Priscila Rodriguez Villacis, de la carrera de Electrónica y Automatización de la facultad de sistemas y telecomunicaciones de la Universidad Estatal Península de Santa Elena, me permito declarar que luego de haber orientado, estudiado y revisado, lo apruebo en todas sus partes y autorizo al estudiante para que inicie los trámites legales correspondientes.

La libertad, 13 de enero del 2024

Ing. Oscar Wladimir Gómez Morales, Mgt

DOCENTE TUTOR

TRIBUNAL DE SUSTENTACIÓN

Ing. Ronald Humberto Rovira Jurado Ph. D.

DIRECTOR DE LA CARRERA DE ELECTRÓNICA Y AUTOMATIZACIÓN

Ing. Oscar Gómez Morales, Mgt.

DOCENTE TUTOR.

Ph.D. Bremnen Marino Véliz Noboa

DOCENTE ESPECIALISTA

Ing. Luis Enrique Chuquimarca Jimenez,Msc

DOCENTE GUIA UIC

Ing. Corina Gonzabay De La A, Mgt

SECRETARIA

RESUMEN

Esta tesis propone el diseño e implementación de una estación meteorológica portátil con capacidades de monitoreo con tecnología IoT para la recopilación y análisis de datos climáticos. La investigación se fundamenta en antecedentes internacionales y nacionales sobre la importancia de las estaciones meteorológicas en sectores como la agricultura, la generación eléctrica y la gestión de riesgos climáticos. La propuesta está diseñada para abordar problemas críticos como la falta de sincronización temporal, la transmisión de datos limitada y la necesidad de análisis predictivos más avanzados.

Se emplea sensores que evaluaran factores climatológicos como la temperatura, la humedad, la velocidad y la dirección del viento, además de la radiación UV. Estos datos son recopilados por un microcontrolador ESP32, mediante la conexión Wi-Fi se comunica a plataformas como Arduino IoT Cloud e IFTTT. Los datos adquiridos se emplean para enviar mensajes de alertas automatizadas que ayudan a prever situaciones meteorológicas inestables, mejorando el tiempo de reacción y planificación en entorno académico.

Además, el sistema emplea el modelo de red neuronal LSTM para examinar datos pasados y anticipar sucesos climáticos futuros, mejorando así la exactitud en la predicción de acontecimientos meteorológicos. Las redes LSTM son entrenadas con información recolectada por los sensores, utilizando datos en tiempo real prediciendo fenómenos como fuertes lluvias, sequías o cambios drásticos de temperatura.

La estación meteorológica integra hardware y software para dar una solución que no solo este centrada en las variables climáticas, más bien agiliza la predicción, ayuda a reducir riesgos medioambientales.

Palabras claves: Estación meteorológica portátil, Monitoreo IoT, Sensores climatológicos, Red neuronal LSTM, Análisis predictivo, ESP32, Alertas automatizadas.

ABSTRACT

This thesis proposes the design and implementation of a portable weather station with monitoring capabilities with IoT technology for the collection and analysis of climate data. The research is based on international and national precedents on the importance of weather stations in sectors such as agriculture, electricity generation and climate risk management. The proposal is designed to address critical issues such as lack of time synchronization, limited data transmission, and the need for more advanced predictive analytics.

Sensors are used to evaluate climatological factors such as temperature, humidity, wind speed and direction, as well as UV radiation. This data is collected by an ESP32 microcontroller, through the Wi-Fi connection it communicates to platforms such as Arduino IoT Cloud and IFTTT. The acquired data is used to send automated alert messages that help to predict unstable weather situations, improving reaction and planning time in academic environments.In addition, the system employs the LSTM neural network model to examine past data and anticipate future weather events, thereby improving accuracy in predicting weather events. LSTM networks are trained with information collected by sensors, using real-time data predicting phenomena such as heavy rainfall, droughts or drastic changes in temperature.

The weather station integrates hardware and software to provide a solution that is not only focused on climatic variables, but rather speeds up prediction, helping to reduce environmental risks.

Keywords: Portable weather station, Iot monitoring, Climatological sensors, LSTM neural network, Predictive analysis, ESP32, Automated alerts.

DECLARACIÓN

El contenido del presente trabajo de titulación es de mi responsabilidad; el patrimonio intelectual del mismo le pertenece a la Universidad Estatal Península de Santa Elena.

Rodriguez V

Lisbette Priscila Rodriguez Villacis

Autora

ÍNDICE GENERAL

AGRADECIMIENTO I
DEDICATORIA II
APROBACIÓN DEL TUTOR III
TRIBUNAL DE SUSTENTACIÓN IV
RESUMEN V
ABSTRACT VI
DECLARACIÓN VII
ÍNDICE GENERAL VIII
ÍNDICE DE FIGURAS XI
ÍNDICE DE TABLAS XIII
CAPÍTULO I 1
1. FUNDAMENTACIÓN. 1
1.1 Antecedentes 1
1.2 Descripción del proyecto 3
1.2.1 Implementación de sensores y recopilación de datos 3
1.2.2 Desarrollo del sistema de monitoreo 4
1.2.3 Integración de alarma y respuesta automatizada. 4
1.3 Objetivos 4
1.3.1 Objetivo General 4
1.3.2 Objetivos específicos. 4
1.4 Justificación 5
1.5 Alcance del proyecto 6
1.6 Metodología 7
1.6.1 Investigación de revisión sistemática 7
1.6.2 Investigación experimental controlada 7
1.6.3 Investigación comparativa 7
1.6.4 Investigación de cuasi experimental 7
CAPÍTULO II 8
2 PROPUESTA 8
2.1 Marco Contextual 8
2.2 Marco conceptual 9
2.2.1 Estaciones meteorológicas 9

2.2.2 Tipo de estaciones meteorológicas 9
2.2.3 Variables meteorológicas 10
2.2.4 Internet de las cosas 12
2.2.5 Plataforma de monitoreo 12
2.2.6 Plataformas en la nube 13
2.2.7 Protocolo de comunicación MQTT 15
2.2.8 Protocolo de comunicación UART 16
2.2.9 Procesamiento y Análisis 16
2.2.10 Redes Neuronales Artificiales 16
2.2.11 Redes neuronales recurrentes 17
2.2.12 Entrenamiento de la red neuronal artificial 17
2.2.13 Arquitectura de la LSTM 18
2.2.14 Funciones de error y evaluación 19
2.2.15 Métodos de imputación 20
2.2.16 Redes de área personal inalámbrica 20
2.3 Comunicación Serial 22
2.3.1 Protocolo RS-232 23
2.3.2 Protocolo Serial Peripheral Inferface 23
2.3.3 Inter Integrated Circuit 23
2.4 Marco teórico 24
CAPÍTULO III 25
3 DESARROLLO DE LA PROPUESTA 25
3.1 COMPONENTES DE LA PROPUESTA 25
3.1.1 Componentes físicos. 25
3.1.2 Componentes lógicos 33
3.2 DISEÑO DE LA PROPUESTA 35
3.2.1 Implementación del sistema de adquisición de datos de la estación meteorológica 36
3.2.2 Configuración de elementos lógicos. 37
3.2.3 Montaje de estación meteorológica portátil 39
3.2.4 Datos obtenido en Arduino Cloud 40
3.2.5 Conexión electrónica 41
3.2.6 Datos en Google drive 42
3.2.7 Configuración de la red neuronal 42

3.3.8 Construcción del Modelo 44
3.3.9 Matriz correlación de las variables 49
3.3.10 Entrenamiento del modelo de percepción 50
3.2.11 Resultados de las variables de la estación meteorológica 51
3.2.12 Pruebas y resultados de la predicción en Python 53
3.3.12 METRICAS 57
CAPÍTULO IV 60
4 CONCLUSIONES Y RECOMENDACIONES 60
4.1 Conclusiones 60
4.2 Recomendaciones 61
ANEXOS 62
ESTRUCTURA FINAL DE LA TESIS 62
Anexo 1 programación ESP32-ArduinoCloud 63
ANEXO 2 ENTRENAMIENTO TEMPERATURA 70
ANEXO 3 ENTRENAMIENTO HUMEDAD 75
ANEXO 4 ENTRENAMIENTO VELOCIDAD DEL VIENTO 80
ANEXO 5 ENTRENAMIENTO RADIACIÓN UV 84
ANEXO 6 MATRIZ CORRELACIÓN 89
Bibliografía 92

ÍNDICE DE FIGURAS

Figura 1. Bosquejo planteado para la implementación de la estación de monitoreo y alarma meteorológico portátil con integración de tecnología IoT. 3
Figura 2. Información meteorológica 9
Figura 3. Internet de las cosas (IoT) 12
Figura 4. AWS IoT 13
Figura 5. Arquitectura Cloud IoT Core 14
Figura 6.Arquitectura según las redes recurrentes 17
Figura 7. Arquitectura de la LTSM 18
Figura 8. Redes de área personal inalámbrica (WPAN) 20
Figura 9. Diagrama de una red WiMAX 21
Figura 10. Redes de telefonía móvil 22
Figura 11. Sensor Termómetro e higrómetro 26
Figura 12. Sensor velocidad del viento 27
Figura 13. Sensor dirección del viento -Veleta 28
Figura 14. Sensor UV 29
Figura 15. Microcontrolador ESP32 31
Figura 16. Panel solar 32
Figura 17. Conexiones físicas y lógicas 36
Figura 18. Flujo grama de la obtención de datos a drive 37
Figura 19. Programación del Esp32 en Arduino cloud 38
Figura 20 montaje de estación meteorológica 40
Figura 21 datos grabados en google drive 42
Figura 22 Arquitectura de la red 47
Figura 23. datos exploradores de variables en Python 47
Figura 24 datos exploradores de variables en Python 48
Figura 25 Matriz correlación de las variables meteorológicas 49
Figura 26 entrenamiento del modelo 51
Figura 27 datos de la temperatura 51
Figura 28 datos de la velocidad del viento 52

Figura 29 Datos de la humedad..... 52
Figura 30 datos de la Radiación UV 53
Figura 31 Predicciones y Valores Reales de Temperatura 54
Figura 32 Comparación de Predicciones y Valores Reales de Radiación UV……..…68**¡Error! Marcador no definido.**
Figura 33 Comparación de Predicciones y Valores Reales de Velocidad del Viento 56
Figura 34 Comparación de Predicciones y Valores Reales de Humedad..... 56
Figura 35Métrica de la variable temperatura 57
Figura 36 Métrica de la variable Humedad..... 58
Figura 37Métrica de la variable velocidad del viento 58
Figura 38 Métrica de la variable Radiación UV 59

ÍNDICE DE TABLAS

Tabla 1 Compuertas de la LSTM 18
Tabla 2 Sensor Termómetro e higrómetro 26
Tabla 3 Sensor de velocidad del viento 27
Tabla 4 Sensor dirección del viento 28
Tabla 5 Radiacion UV 29
Tabla 6 Especificaciones del microcontrolador ESP32 31
Tabla 7 especificación del panel solar 32
Tabla 8 tabla con los eventos recibidos 41
Tabla 9 Bibliotecas para la construcción del modelo LSTM 45
Tabla 10. Parametros configurados. 45
Tabla 11 Arquitectura de la red LSTM 46

CAPÍTULO I

1. FUNDAMENTACIÓN.

1.1 Antecedentes

La ONU en el 2020, ha implementado diversas ideas innovadoras para los sistemas de alerta temprana en sectores vulnerables, se desarrolló un programa especial para implementar estaciones climáticas y automatizadas, para el sector de la agricultura, ahora las personas pueden obtener información detalladas [1].

Dentro de los requerimientos técnicos asignados por el IDEAM para siete estaciones hidrometereológicas automáticas, presentaron propuestas en el área circundante al lago Tota en Boyacá. En primer lugar, resalta la necesidad de contar con una alimentación eléctrica regulada que asegure un suministro constante y estable de energía, es fundamental para el funcionamiento continuo de las estaciones. Se requiere la implementación de un dispositivo que facilite la sincronización periódica y automática de la fecha y hora, ya sea a través de capacidades internas de la estación o mediante la interacción con equipos de comunicación externos a los que estén conectadas. De esta forma garantiza que los datos recopilados estén temporalmente alineados y sean fácilmente comparables y analizables [2].

Empresa Electro Generadora del Austro, se planifico un estudio de necesidad para la toma de datos meteorológicos, en las sucursales hidroeléctricas llevándolos a tener un convenio con ETAPA EP sus estaciones meteorológicas miden parámetros como: temperatura ambiental, humedad, relativa, velocidad de viento, lluvia, etc., los datos que son recopilados se toman para poder planificar programaciones de energía. ELECAUSTRO S.A en su parque de generación se ha aumentado las centrales hidroeléctricas, en la zona Minecentral de Guacaleo se ve la necesidad de implementar una estación meteorológica para tomar datos físicos [3].

En la propuesta de diseño de la red hidrometereológica nacional, el IDEAM informa que la mayoría de las estaciones automáticas, aproximadamente el 74%, utilizan el Satélite Geoestacionario Operacional Ambiental (GOES) del programa estadounidense del National

Weather Service (NWS) de la National Oceanic and Atmospheric Administration (NOAA) para transmitir datos. Sin embargo, debido que este servicio es gratuito para países como Colombia, los datos recibidos a través de este satélite pueden experimentar retrasos de hasta una hora. Un porcentaje menor, el 9% emplea el servicio general de paquetes vía radio (GPRS) con intervalos de una hora para la transmisión de datos, mientras que el 12% de las estaciones automáticas no cuentan con un sistema de transmisión, almacenando la información en la memoria interna del sensor. Además, el 5% de las estaciones corresponden a estaciones sinópticas en aeropuertos, que transmiten datos con una frecuencia de 2 minutos y son cruciales para el servicio de meteorología aeronáutica prestado por el IDEAM [4].

El estudio de la hidrología en la cuenca del Rio Claro, Colombia, además del estudio de los impactos socioeconómicos relacionados con el sistema glacial, se hizo más sencillo con la utilización de una estación meteorológica automática por satélite, resaltado en el reporte de actividades glaciológicas de la Sierra Nevada el Cocuy y el volcán Nevado Santa Isabel. No obstante, existen retos importantes vinculados con el mantenimiento y la confiabilidad del equipo. Se señaló que las deficiencias en los sensores y en la transmisión de datos dificultaron el procesamiento y estudio de la información recolectada [5].

En Ecuador desde el año 2023 se ha presentado niveles altos y extremos de radiación ultravioleta (UV) si bien su escala se divide en cinco categorías como: Bajo, Moderado, alto, muy alto, extremadamente alto. Exponerse por mucho tiempo es perjudicial para la salud. Según informa el Instituto Nacional de Meteorología e Hidrología, la mayoría de las provincias sufren de radiación UV alta y muy alta, ninguna se encuentra en el punto moderado. Los informes climatológicos de la INAMHI a nivel general ocasionan que algunas empresas acudan a entidades particulares para construir sus propias estaciones [6].

Coca Codo Sinclair cuenta con un sistema de alerta temprana (SAT), que permite prevenir afectaciones a la casa de máquinas, crecidas del rio Coca. Un sistema moderno de monitoreo en tres estaciones hidrométricas incluye la instalación de equipos electrónicos que mide el nivel, volumen de sólidos y líquidos, también incluye alertas tempranas para las inundaciones empleando sensores de nivel de agua y luces para emitir señal de alertas [7].

1.2 Descripción del proyecto

Este proyecto cuenta con la implementación de una estación de monitoreo y alarma meteorológico portátil con integración de tecnología IoT. En este sistema se establece el uso de protocolos de comunicación MQTT. A continuación, el proceso se divide en las siguientes etapas:

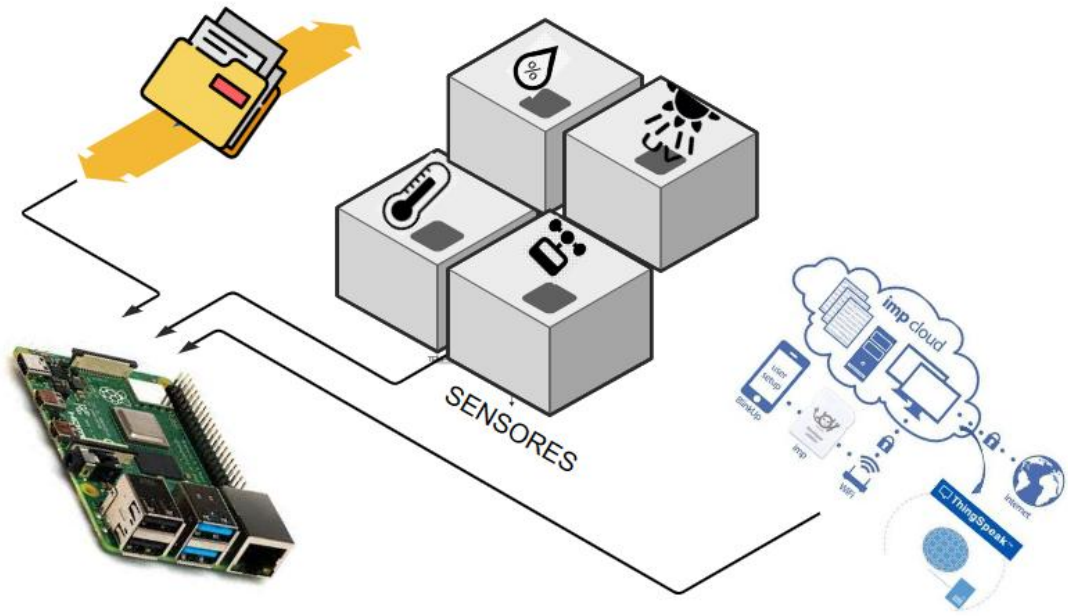


Figura 1. Bosquejo planteado para la implementación de la estación de monitoreo y alarma meteorológico portátil con integración de tecnología IoT.

Fuente: L. Rodriguez,2024

1.2.1 Implementación de sensores y recopilación de datos

En la etapa uno contamos con los dispositivos para optimizar la implementación de la estación meteorológica, está compuesto por un subsistema de sensores los cuales serán encargados de las variables físicas como (temperatura, humedad, velocidad del viento, rayos ultravioletas). Estos datos que recibimos de los sensores serán recopilados por un microcontrolador, se toma en cuenta su compatibilidad con los sensores para manejar datos correctos y confiables.

1.2.2 Desarrollo del sistema de monitoreo

En la segunda etapa implementamos un sistema de monitoreo remoto que permitirá a las autoridades del campus acceder a los datos meteorológicos desde cualquier dispositivo en tiempo real, desarrollando una interfaz de usuario accesible de usar. El sistema puede ser visto desde cualquier dispositivo con conexión a internet, utiliza tecnología para procesar datos con precisión asegurando que la información sea lo más precisa.

1.2.3 Integración de alarma y respuesta automatizada.

En la última etapa se diseña un sistema de detección de condiciones peligrosas para generar la funcionalidad de una alarma que se activará automáticamente en caso de detectarse algún indicio de riesgo, como fuertes lluvias, cambios bruscos de temperatura, inundaciones. La disponibilidad de la información permitirá una planificación más eficiente en eventos, actividades académicas y procedimiento diario del campus. Además, se usará unas redes neuronales para analizar los datos con el fin de predecir eventos futuros.

1.3 Objetivos

1.3.1 Objetivo General

Diseñar un sistema de comunicación con acceso remoto a la nube para monitorear las variables físicas climáticas, activación de alarmas y análisis predictivo con inteligencia artificial, con el fin de mejorar la capacidad de anticipación y respuesta ante eventos meteorológicos.

1.3.2 Objetivos específicos.

- Investigar tres variables ambientales que se vinculen con el cambio climático para permitir una planificación eficiente ante posibles nuevos riesgos ambientales.
- Implementar la plataforma IoT para la recopilación de datos desde la estación meteorológica para el monitoreo de las variables físicas climáticas.

- Desarrollar una interfaz hombre-maquina (HMI) para la supervisión de los sensores de manera local, en la estación meteorológica.
- Integrar un sistema de entrenamiento para mejorar la capacidad de anticipación ante eventos climáticos, con los datos recopilados con la estación meteorológica.
- Pruebas y validación del sistema IoT y el modelo de entrenamiento neuronal.

1.4 Justificación

El estudio de variables climatológicas muestra un gran impacto a la comunidad científica por su influencia en sistemas naturales [8]. Las estaciones meteorológicas son de ayuda para predecir el clima, hasta relacionarlos con investigaciones y estudios epidemiológicos. Se destaca la detección de riesgos climáticos e investigaciones al relacionarse con una contaminación atmosférica. Sus variables físicas pueden ser aplicadas a programas de conservación ambiental, para ser aplicada en planes de manejo ambiental.

AEBE, Asociación de Exportadores de Banano del Ecuador, buscan una solución para que se les permita integrar una red de estaciones meteorológicas que permitan obtener monitorear y conta con la información del clima, e indicadores que podrían beneficiar a la producción del banano [9]. Por lo tanto, este proyecto propone establecer un sistema de monitoreo para las variables físicas climatológicas, los parámetros de suma importancia como; temperatura, humedad, velocidad del viento.

El estudio y observación del incremento de altas temperaturas, radiación, sequias y tormentas se ha convertido en un método preventivo, debido a la eficaz implementación de una alarma con el fin de mejorar la respuesta ante dichos fenómenos.

Monitorear las variables climáticas es necesario para prever el estiaje en las hidroeléctricas, puesto que estas plantas dependen en gran medida del caudal de los ríos para generar electricidad. En el tiempo de sequía prolongada, como suele ocurrir durante el estiaje, el caudal de los ríos puede disminuir significativamente, lo que reduce la capacidad de generación de energía hidroeléctrica. La energía creada por una planta hidroeléctrica es la principal fuente de abastecimiento eléctrico del país. El 60% de la electricidad proviene de las plantas hidroeléctricas, en los últimos años el país ha experimentado dificultades en su sector eléctrico debido a sequias severas.

Dado los desafíos que ha presentado el país se busca afrontarlos incluyendo sistemas de monitoreos que analicen las variables climáticas y su respuesta predictiva, que ayudarían a reducir el impacto que produce efecto del estiaje y asegurar un abastecimiento eléctrico más constante.

El sistema que se propone se implementará en la Universidad Estatal Península de Santa Elena en donde se llevará a cabo las pruebas pertinentes para alcanzar los objetivos del proyecto.

1.5 Alcance del proyecto

La solución IoT cuenta con la tecnología inalámbrica, que nos ayudara a recopilar los datos de las variables físicas climatológicas, un algoritmo inteligente que es encargo de predecir y diagnosticar fallos. La implementación de esta solución IoT es de nivel básico y de forma experimental donde el propósito es demostrar su potencial en el ámbito tecnológico. La funcionalidad de cada bloque del sistema. Tanto a nivel de hardware / software es la siguiente:

- La estación está equipada con sensores especializados en la medición de variables meteorológicas como temperatura, humedad, velocidad del viento. Estos sensores receptan continuamente los datos ambientales y los transmiten de forma inalámbrica a través de la plataforma IoT.
- En esta fase incluye la recopilación de datos que son adquirido de los sensores y el microcontrolador, de esta forma habiendo tomado la información de las variables se las envía a la plataforma IoT para su respectivo procesamiento.
- El Gateway, por su parte concede la comunicación entre el nodo final y la plataforma IoT. El microcontrolador desempeña un papel vital en la transmisión de datos, garantizando que la información climática se transmita a la plataforma de seguimiento garantizando la recepción de los datos de forma precisa.
- La plataforma de Arduino Cloud IoT se ocupa de la interfaz de usuario, facilitando el acceso y estudio de la información climática recolectada por la estación de meteorológica. Esta plataforma enviara mensajes de alertas basadas en los umbrales predefinidos.

1.6 Metodología

1.6.1 Investigación de revisión sistemática

Al realizar un análisis sistemático de la literatura, se facilita la recolección, análisis y síntesis de la evidencia existente con relación al asunto de la estación meteorológica portátil y la incorporación de la tecnología IoT. Esto asiste en la identificación de fisuras en el saber existente, además de ajustar tu estudio dentro del marco teórico y conceptual establecido en la disciplina.

1.6.2 Investigación experimental controlada

Al implementar la estación meteorológica portátil en zonas descubiertas, se evidencia la precisión y lo que puede afectar al rendimiento de los datos producidos por las variables meteorológicas. Si se observan y manipulan las variables el efecto que tendrá permitirá generar deducciones estables.

1.6.3 **Investigación comparativa**

El estudio comparativo involucra analizar y comparar diferentes tipos de estaciones meteorológicas portátiles disponibles en el mercado, así como también evaluar sus características, rendimiento y precisión. Permite identificar las fortalezas y debilidades de cada sistema lo que lleva a recomendaciones para la selección y mejora de la estación meteorológica portátil a implementar.

1.6.4 **Investigación de cuasi experimental**

El estudio de desarrollo se enfoca en el proceso de diseño. Se definen las necesidades del diseño, elegimos sensores y materiales auxiliares, llevamos a cabo ensayos en el prototipo. Facilita el registro del proceso de desarrollo y verifica la eficacia del prototipo que se obtiene.

CAPÍTULO II

2 PROPUESTA

2.1 Marco Contextual

El proyecto nos permite implementar una estación de monitoreo y alarma meteorológico portátil con integración de tecnología IoT para la Facultad de Sistemas y Telecomunicaciones de la Universidad Estatal Península de Santa Elena. En este sistema se considera los dispositivos meteorológicos apropiados para el uso en campos extensos.

Para el desarrollo del sistema se investigarán proyectos relacionados con la tecnología IoT. Esta perspectiva acepta ampliar pequeños cambios en las condiciones meteorológicas, favoreciendo una detección precisa y temprana de las variables climatológicas.

Es importante destacar que las mediciones se realizaran en un rango determinado para las condiciones ambientales y en diferentes ubicaciones de la facultad asegurando, que los datos que se recolecten cumplan con lo necesario para funcionamiento de nuestra estación. Se deberá garantizar un entorno controlado durante las pruebas, posicionamiento adecuado de los sensores, además de cumplir con las indicaciones para no presentar problemas en la recolección y análisis de datos meteorológicos.

La estación comprende herramientas como microcontroladores y sensores de humedad, temperatura, velocidad del viento y radiación UV que permitan el monitoreo de las variables. La arquitectura en la nube se basa en Arduino Cloud que nos ayuda a almacenar datos, la implementación de la interfaz gráfica para supervisar y predecir condiciones climáticas. Con los datos obtenidos el sistema propuesto identificará patrones, el modelo de predicción obtendrá correctamente indicios de variaciones climáticas.

La implementación de estaciones meteorológicas portátiles de monitoreo y alerta permiten a las autoridades y al público recibir información oportuna y precisa. Esto mejorará la capacidad de reacción y planeación de estrategias preventivas a condiciones climáticas adversas contribuyendo a la seguridad y el bienestar públicos.

2.2 Marco conceptual

2.2.1 Estaciones meteorológicas

Las estaciones meteorológicas se realizan con diferentes fines. La realización del análisis para pronósticos y alertas sobre la agudeza de fenómenos naturales, se utilizan además para ciertas operaciones locales (aeródromos, en la operación de cierta maquinaria en la construcción, etc.), para pronósticos hidrológicos y agrometeorológicos, así como para investigación en ciencias de la atmósfera. A continuación, la fig.1.2 nos muestra la información meteorológica.

Figura 2. Información meteorológica

Fuente: [10]

2.2.2 Tipo de estaciones meteorológicas

2.2.2.1 Estaciones meteorológicas automáticas

Este tipo de estaciones vienen a complementar la red de observaciones convencionales. Lo hacen proporcionando datos de lugares de difícil acceso, como en estación dotadas de personal, efectuando observaciones adicionales. Abarcan todo tipo de sensores que miden los diferentes elementos meteorológicos. Así, por ejemplo, en una estación meteorológica

automática se incluyen sensores que registran la dirección, velocidad del viento, temperatura del aire, humedad, la radiación solar y otros.

También dependen de diversos factores físicos y psicológicos típicos del observador, como sus hábitos de los objetos y sus posiciones y la adaptación de sus ojos a las condiciones ambientales nocturnas y en este caso el viento, se establece como una variable de "valor promedio", es decir este ayuda al observador leer datos que están en una pantalla analógica, y este resultado es valor promedio que permite un cálculo sobre esta área y que ayuda como aporte para las estaciones meteorológicas [11].

2.2.2.2 Radares Meteorológicos

Sus principales componentes son el transmisor, la antena y el receptor; el transmisor, la antena y el receptor; el transmisor genera cortos impulso de energía. El objetivo fundamental de este tipo de planta es determinar la distancia y naturaleza o comportamientos.

El radar meteorológico se ha convertido en un instrumento de observación esencial para detectar diversos tipos de fenómenos atmosféricos y sus procesos de desarrollo. Se utilizan para detectar el desarrollo de nubes, medir la lluvia, control de huracanes, prevención de granizo, para determinar perfiles de viento. Existen también radares para localizar tormentas y perturbaciones atmosféricas [12].

2.2.3 Variables meteorológicas

Las variables meteorológicas han crecido a lo largo del tiempo ya que la información que brindas las estaciones que monitorean el entorno es vital para el diario vivir, así mismo con estas variables determinan cada cierto tiempo el automatizar las estaciones meteorológicas. Dentro de la industria existen parámetros importantes de medición como es la temperatura y su almacenaje de datos.

2.2.3.1 Velocidad del viento

La velocidad del viento se mide generalmente utilizando métodos que detectan el movimiento del aire. Un enfoque común es medir la rotación de un de un dispositivo mecánico que se mueve con el viento, traduciendo esta rotación en una velocidad. Otra técnica es el uso de ultrasonidos que calculan la velocidad del viento midiendo el tiempo

que tardan las ondas sonoras en desplazarse entre diferentes puntos. La velocidad del viento es crucial para entender patrones meteorológicos, predicciones de tormentas y evaluar condiciones para actividades como la aviación y la navegación.

2.2.3.2 Temperatura

La temperatura se mide detectando cambios en la resistencia de materiales sensibles al calor o por la radiación infrarroja emitida por el objeto que se mide. Estos cambios se convierten en valores de temperatura utilizando métodos físicos y matemáticos. La medición de la temperatura es esencial para aplicaciones como el monitoreo climático, la planificación agrícola y la optimización de recursos energéticos.

2.2.3.3 Humedad relativa

La humedad relativa se mide comparando la cantidad de vapor de agua en el aire con la cantidad máxima de vapor de agua que el aire puede contener a esa temperatura. Esto se logra utilizando materiales que cambian sus propiedades eléctricas bajo la influencia de la humedad o mediante sensores capacitivos y resistivos. La humedad relativa es esencial para pronosticar la formación de nubes, niebla y precipitaciones y para el control de calidad en industrias sensibles a la humedad.

2.2.3.4 Radiación UV

La radiación ultravioleta se mide determinando la intensidad de la radiación que alcanza la superficie terrestre. Para esto, se utilizan sensores especializados que responden a diferentes rangos de longitudes de onda UV, permitiendo obtener mediciones precisas de su intensidad. Esta información es fundamental para evaluar los riesgos que se pueden provocar a la salud humana, como la resequedad de la piel y el aumento del riesgo de cáncer de piel. Incluso, resulta crucial para estudios ambientales y de energía solar.

2.2.4 Internet de las cosas

El internet de las cosas (IoT) como sus siglas indican en ingles define la conectividad de dispositivos físicos a través de Internet ayuda facilitando la comunicación y la recopilación de los datos entre sí. Introduce un cambio en la calidad de vida de las personas dando variedad de oportunidades para acceder a datos, servicios para la educación, entre demás campos. Un objeto es inteligente siempre y cuando se tome en cuenta el ambiente, es una representación de cómo se comporta y de la interacción con el usuario [13].

2.2.5 Plataforma de monitoreo

Se dispone de sensores que adquieren datos del entorno, estos valoran las condiciones del clima, su información recopilada ayuda a mostrar el comportamiento mediante graficas donde se pueden observar los picos de cambios. La plataforma se configura para enviar mensajes de alerta dado el umbral que se coloque en su análisis y monitoreo en tiempo real.

Los protocolos de comunicación como MQTT y HTTP muestran la transmisión de datos de forma eficiente y segura. El mantenimiento regular de los sensores es fundamental para prevenir fallos y prolongar su vida útil junto con la batería. [14].

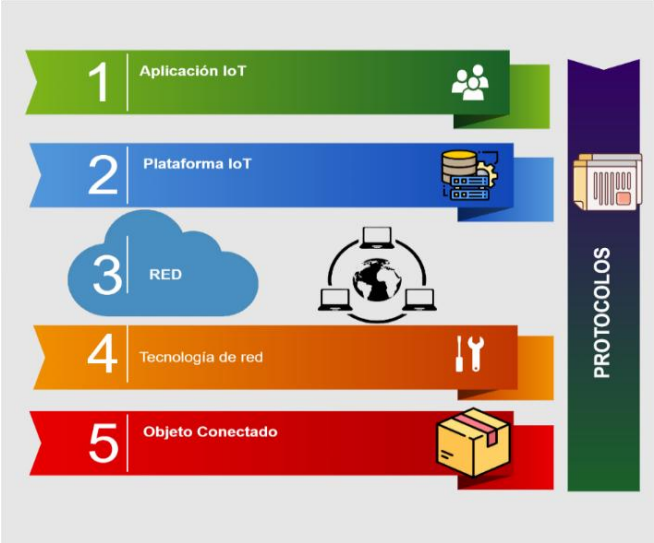


Figura 3. Internet de las cosas (IoT)

Fuente: Autor

2.2.6 Plataformas en la nube

Las plataformas en la nube son aquellos servicios que nos brindan almacenamiento de datos y aquellos recursos que se comparten a través de internet, ciertas plataformas necesitan pagar solo por su acceso, sin embargo, existen las gratuitas.

2.2.6.1 Servicios web de Amazon IoT

- AWS IoT Core: Es un servicio de la nube, que permite conectar dispositivos para que interactúen con otros dispositivos de manera segura.
- AWS IoT Analytics: Crea aplicación de IoT analiza los datos de IoT de manera que recopila esa información, puede almacenar los datos. Cuenta con aprendizaje autónomo y factura según el volumen de datos que son procesados en gigabytes.
- AWS IoT Greengrass: Es un software que permite crear y administrar software de dispositivos locales estos recopilan datos y reaccionan de manera autónoma para poder comunicarse de forma segura con dispositivos.

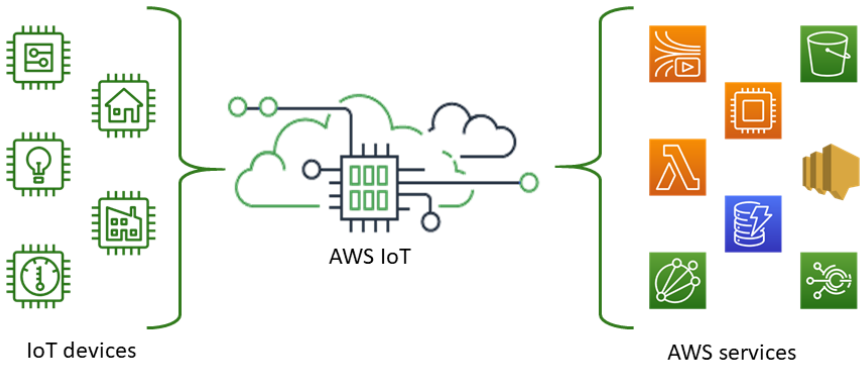


Figura 4. AWS IoT

Fuente: [15]

2.2.6.2 Google Cloud IoT

Cloud IoT Core: Servicio completamente administrado que permite conectar, administrar y obtener datos de dispositivos IoT de manera global.

Cloud IoT Edge: Es conocido por ser un modelo informático que procesa datos de baja y alta procedencia del Internet de las cosas, recopila datos de sensores ya sea para sistemas de seguridad o trabajos en la nube.

BigQuery: Dado que esta plataforma procesa datos de Google cloud, a su vez ayuda a incrementar los datos que se obtengan es completamente administrada para la IA.

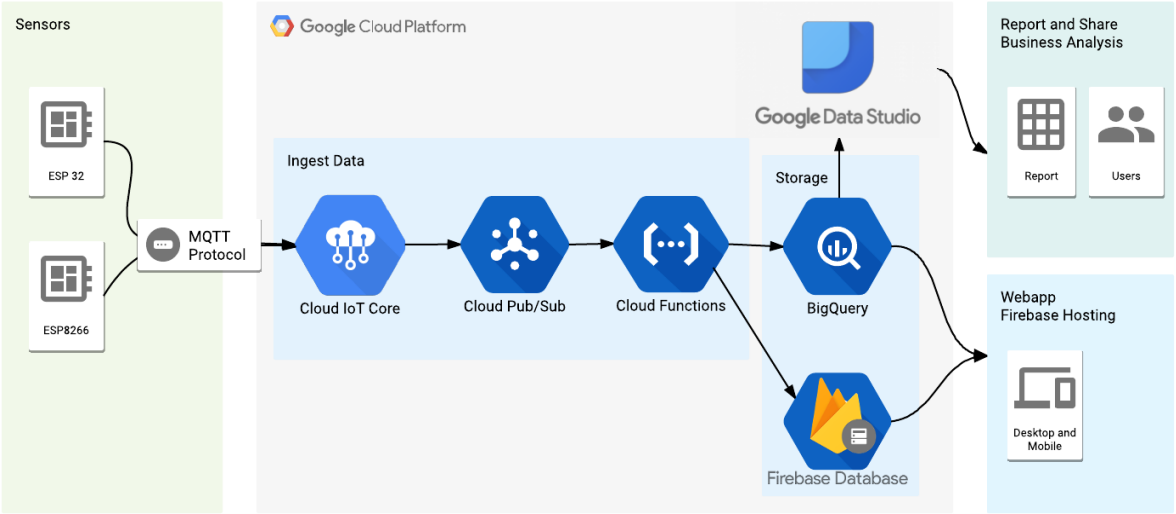


Figura 5. Arquitectura Cloud IoT Core

Fuente: [16]

2.2.6.3 Thingspeak

ThingSpeak es una plataforma en la nube diseñada para el Internet de las Cosas (IoT) que permite recopilar, almacenar, analizar y visualizar datos en tiempo real de dispositivos conectados. Es ampliamente utilizada en proyectos de IoT y en aplicaciones donde se monitorean datos provenientes de sensores, como en la meteorología, la automatización del hogar, y la agricultura.

Una de las características clave de ThingSpeak es la capacidad de almacenar datos en canales. Cada canal actúa como un espacio de almacenamiento específico para un conjunto

de datos relacionados y puede configurarse con múltiples campos que representan diferentes tipos de información (por ejemplo, temperatura, humedad, radiación UV, etc.).

Esto permite que los usuarios organicen y administren datos de manera eficiente según el tipo de medición.

ThingSpeak también ofrece una opción para visualizar los datos en gráficos en tiempo real, lo que facilita el seguimiento y la comprensión de tendencias a medida que se van generando los datos. Esta funcionalidad es útil para analizar patrones, detectar anomalías o simplemente observar cambios a lo largo del tiempo.

2.2.6.4 API

Una API (Interfaz de Programación de Aplicaciones) es un conjunto de reglas y definiciones que permiten que diferentes programas y aplicaciones se comuniquen entre sí. En términos simples, una API actúa como un intermediario que permite que una aplicación interactúe con otras aplicaciones o servicios, facilitando el intercambio de datos y funcionalidades sin necesidad de que los desarrolladores conozcan el funcionamiento interno de esos sistemas.

Las APIs permiten que aplicaciones y sistemas se conecten de forma remota para solicitar información, realizar acciones, o acceder a servicios específicos. Por ejemplo, cuando un usuario utiliza una aplicación meteorológica para ver el clima, esta aplicación puede usar una API para obtener datos de un servicio meteorológico en la nube, procesarlos y mostrarlos en el dispositivo del usuario. La API simplifica este proceso al proporcionar instrucciones claras sobre cómo se debe realizar la solicitud y qué tipo de respuesta se puede esperar.

Existen diferentes tipos de APIs, como REST (Representational State Transfer), que es una de las más comunes en aplicaciones web y móviles. Las APIs REST permiten que las aplicaciones interactúen a través de solicitudes HTTP, lo que facilita la transferencia de datos entre un servidor y un cliente. Esto es útil para enviar y recibir información de forma eficiente, especialmente cuando se trata de datos en tiempo real.

2.2.7 Protocolo de comunicación MQTT

Es un protocolo abierto de conectividad, deseñado como transporte de mensajería ligera, ideal para aplicar en el Internet de las cosas, que permite la comunicación confiable entre

dispositivos. En el contexto de una estación de monitoreo meteorológico, MQTT se utiliza para enviar los datos recopilados por los sensores a un servidor central o a la nube, donde pueden ser accedidos y analizados en tiempo real. [17]

2.2.8 Protocolo de comunicación UART

Maneja la comunicación serie. UART (Universal Asynchronous Receiver-Transmitter) es otro componente clave en la capa de comunicación. UART es un protocolo de comunicación serial utilizado para la transmisión de datos entre el microcontrolador y otros módulos periféricos, como los módulos de comunicación inalámbrica o dispositivos de almacenamiento. UART permite una transmisión de datos sencilla y efectiva, facilitando la integración de diversos componentes en la estación de monitoreo. [18]

2.2.9 Procesamiento y Análisis

En la capa de procesamiento y análisis, se realiza un procesamiento de señales con el propósito de asegurar la precisión de los datos recopilados por los sensores en la estación de monitoreo meteorológico. Este procedimiento está destinado a garantizar la corrección y la exactitud de los datos, permitiendo obtener información significativa y fiable.

Los resultados del procesamiento y análisis son visualizados en plataformas de software, que permiten a los usuarios monitorear las condiciones meteorológicas en tiempo real y obtener informes detallados. Estas visualizaciones pueden incluir gráficos, mapas y alertas, facilitando la interpretación de los datos y la respuesta rápida a eventos meteorológicos extremos.

2.2.10 Redes Neuronales Artificiales

Una red neuronal puede ser entrenada utilizando grandes volúmenes de datos históricos sobre condiciones meteorológicas para predecir eventos climáticos extremos, como tormentas o inundaciones. Al analizar variables como temperatura, presión atmosférica, humedad y velocidad del viento, la red neuronal puede generar modelos predictivos que anticipen cambios significativos en el clima, proporcionando alertas tempranas y mejorando la capacidad de respuesta ante emergencias.

2.2.11 Redes neuronales recurrentes

Las redes neuronales recurrentes (RNN), basadas en los trabajos de David Rumelhart en 1986, son una clase de aprendizaje profundo reconocida por su habilidad para procesar y extraer información de datos secuenciales. Esta propiedad las convierte en herramientas esenciales para tareas como el análisis de video, la generación de subtítulos para imágenes, el procesamiento del lenguaje natural (PLN) y el análisis de música. A diferencia de las redes neuronales artificiales tradicionales, que suponen independencia entre los datos de entrada, las RNN son capaces de captar las relaciones secuenciales y temporales presentes en los datos.

Esta limitación se observa especialmente en el procesamiento del lenguaje natural, donde una red multicapa tradicional fallaría al interpretar el lenguaje de manera rígida. En cambio, las RNN, al compartir pesos entre datos secuenciales, son más aptas para tareas como la interpretación de frases en el lenguaje natural. [19]

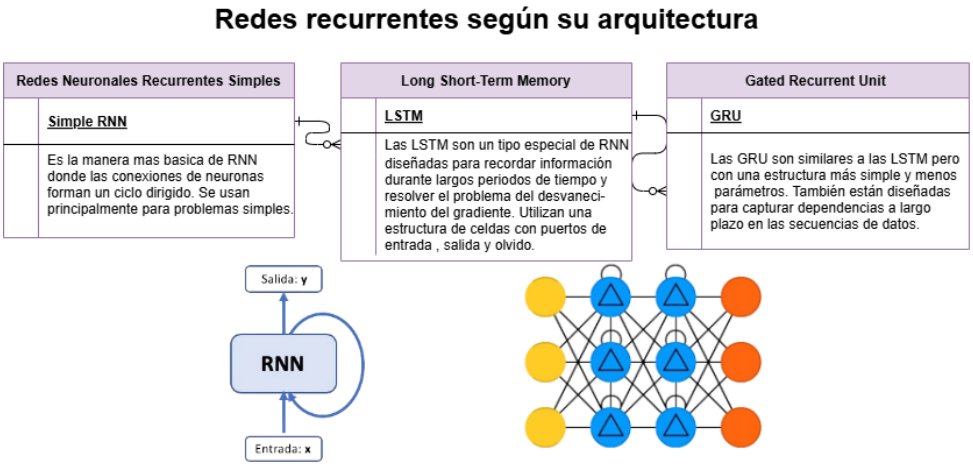


Figura 6.Arquitectura según las redes recurrentes

Fuente: Autor

2.2.12 Entrenamiento de la red neuronal artificial

El aprendizaje supervisado se caracteriza por ser un proceso en el cual el entrenamiento de la red neuronal se realiza bajo la supervisión y control de un agente externo, como un supervisor o maestro. Este agente determina la respuesta que la red debería generar ante una

determinada entrada. El supervisor monitoriza la salida de la red y, en caso de que no coincida con la respuesta deseada, se procede a ajustar los pesos de las conexiones con el objetivo de acercar la salida obtenida a la deseada.

2.2.13 Arquitectura de la LSTM

Diseñada para procesar secuencia de datos, tiene memoria a largo plazo para datos secuenciales, está compuesta por:

Tabla 1 Compuertas de la LSTM

Compuerta de olvido	Información pasada de la memoria que debe eliminarse
Compuerta de Entrada	Determina la nueva información que debe almacenarse
Compuerta de Salida	La información de la memoria se utiliza para la predicción actual.

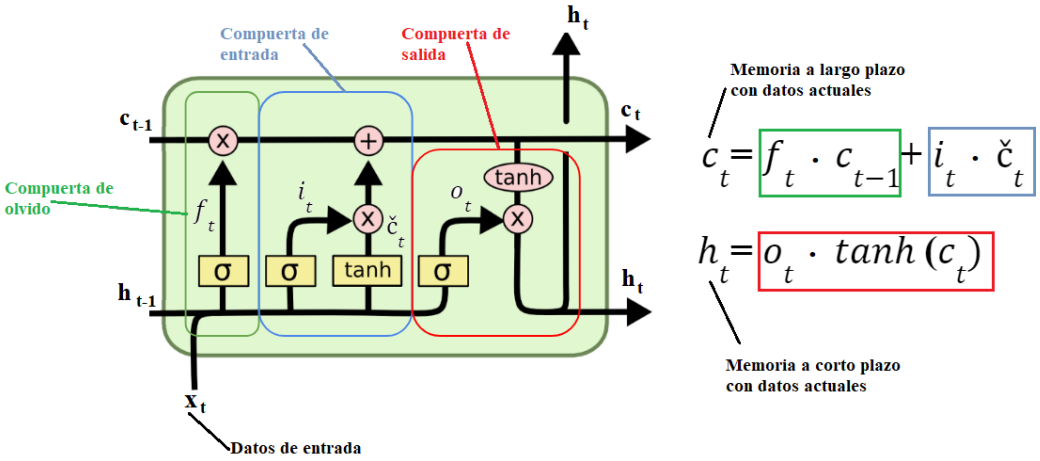


Figura 7. Arquitectura de la LTSM

Fuente: [20]

2.2.14 Funciones de error y evaluación

El error cuadrático medio (MSE), útil para medir la precisión del modelo, a su vez nos lleva a utilizar la raíz de error cuadrático medio (RMSE), interpreta mismas unidades que las variables originales usada para regresión (como son predicciones). En este caso las variables que usamos para la estación meteorológica normalmente como temperatura, radiación son continuas, por lo que es usado el método de regresión.

$$\boldsymbol{RMSE} = \sqrt{\frac{\mathbf{1}}{\boldsymbol{n}}\sum_{\boldsymbol{i=1}}^{\boldsymbol{n}}(\boldsymbol{y_i} - \boldsymbol{F_i})^{\mathbf{2}}} \qquad \textit{Ecuación.1.1}$$

El RSME tiene similitud con el MSE ya que los errores positivos y negativos se compensan entre ellos, por lo que se usa para medir distorsiones con anticipación.

Coeficiente de correlación múltiple (R2)

Indica qué tan bien el modelo explica la variabilidad de los datos. Se utiliza en refresion para evaluar la calidad del ajuste del modelo. Cuando el R^2 es igual a 1, significa que el modelo explica perfectamente toda la variabilidad de la variable objetivo; es decir, las predicciones coinciden exactamente con los valores reales. Un valor de R^2 cercano a 0 indica que el modelo no explica casi nada de la variabilidad en los datos, mientras que un R^2negativo sugiere que el modelo es menos útil que simplemente usar la media de los valores reales como predicción constante [21].

$$R^2 = \frac{\sum_{t=1}^{T}(\hat{y}_t - \bar{y})^2}{\sum_{t=1}^{T}(\hat{y}_t - \bar{y})^2}$$

Ecuación 1.2

Mean Absolute Error

Es una métrica sencilla que calcula el promedio de las diferencias absolutas entre los valores predichos y los reales, lo que me permite evaluar qué tan alejadas están, en promedio, las predicciones del modelo. Al no penalizar los errores grandes de forma excesiva, el MAE resulta útil para tener una visión clara y directa del desempeño del modelo. Un valor más bajo indica que las predicciones están más cerca de los valores reales.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

Ecuación 1.3

2.2.15 Métodos de imputación

Al tener datos meteorológicos con valores faltantes, se manejar métodos de manera que no afecten las predicciones. Existen técnicas para manejar estos datos llamadas:

- Imputación por promedio o mediana: Reemplaza los valores faltantes con el promedio de la columna. Es útil para columnas donde los valores tienen una distribución cercana a la normal.
- Imputación por Forward: Usa el valor que sigue para llenar sus datos faltantes.
- Interpolación: Calcular los valores que faltan basándose en una interpolación lineal.

2.2.16 Redes de área personal inalámbrica

Una WPAN (Red Inalámbrica de Área Personal) es un tipo de red inalámbrica diseñada para cubrir distancias cortas, facilitando la interacción entre dispositivos electrónicos personales como teléfonos móviles, laptops, auriculares inalámbricos y otros equipos portátiles. Estas redes suelen emplear tecnologías como Bluetooth, Zigbee o Near Field Communication

(NFC) para posibilitar el intercambio de datos y la conexión entre dispositivos cercanos.

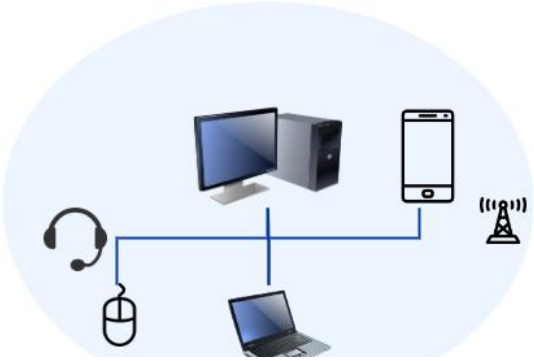


Figura 8. Redes de área personal inalámbrica (WPAN)

Fuente: Autor

2.2.16.1 Redes de área local inalámbrica

Las redes inalámbricas de área local (WLAN) están pensadas para ofrecer conectividad inalámbrica en espacios con un alcance promedio de hasta 100 metros, siendo comunes en hogares, escuelas, salas de computadoras y oficinas. Estas redes se fundamentan en el estándar 802.11 del IEEE y se comercializan bajo el nombre de Wi-Fi. Otros estándares, como HIPERLAN, no lograron la misma aceptación comercial debido a la competencia. El estándar IEEE 802.11 destacó por su facilidad de implementación y su rápida adopción en el mercado.

2.2.16.2 Redes inalámbricas de área metropolitana

Las redes inalámbricas de área metropolitana (WMAN) representan el tercer tipo de redes inalámbricas. Estas redes se desarrollan siguiendo el estándar IEEE 802.16, comúnmente conocido como WiMAX (Worldwide Interoperability for Microwave Access). WiMAX es una tecnología de comunicación con una arquitectura punto a multipunto, diseñada para ofrecer altas velocidades de transmisión de datos en redes inalámbricas de área metropolitana.

Figura 9. Diagrama de una red WiMAX

Fuente: [21]

2.2.16.3 Redes inalámbricas de área amplia

Las redes inalámbricas de área amplia tienen un alcance que supera los 50 kilómetros y generalmente operan en frecuencias con licencia. Estas redes están diseñadas para cubrir grandes territorios, como ciudades o incluso países, utilizando sistemas de satélites o estaciones con antenas administradas por proveedores de servicios de Internet. Las principales tecnologías empleadas para estas redes son la telefonía móvil y los satélites.

2.2.16.4 Red de telefonía móvil

En la red de telefonía móvil, el área de cobertura se organiza en celdas. Cada celda cuenta con un transmisor o estación base ubicado en su centro, encargado de proporcionar servicio dentro de esa celda específica. Los dispositivos móviles se conectan a la estación base, la cual, a su vez, está enlazada a una central de conmutación de telefonía móvil que permite la conexión entre el teléfono móvil y la red telefónica cableada.

Figura 10. Redes de telefonía móvil

Fuente: [21]

2.3 Comunicación Serial

En los sistemas embebidos y dispositivos electrónicos, los microcontroladores emplean protocolos de comunicación para intercambiar datos de forma eficaz. Entre los más

utilizados se encuentran RS-232, SPI e I2C, cada uno con características específicas para gestionar la transferencia de información.

2.3.1 Protocolo RS-232

El protocolo RS-232, uno de los más antiguos, se emplea principalmente para comunicaciones seriales, ya sean unidireccionales o bidireccionales. Es comúnmente utilizado para conectar dispositivos, como computadoras y periféricos, en distancias cortas. Su funcionamiento se basa en la transmisión de datos a través de un único cable, siguiendo un estándar de voltaje que permite la interacción entre dos dispositivos [22]

2.3.2 Protocolo Serial Peripheral Inferface

SPI es un protocolo de comunicación caracterizado por su sincronización y alta velocidad, diseñado para transferir datos entre un microcontrolador y uno o varios dispositivos periféricos.

Utiliza una arquitectura maestro-esclavo donde el dispositivo maestro controla la comunicación y los esclavos responden según las instrucciones. [23]

2.3.3 Inter Integrated Circuit

I2C es un protocolo que permite la comunicación entre múltiples dispositivos a través de dos líneas: una para el reloj (SCL) y otra para los datos (SDA). Este protocolo es especialmente útil para conectar varios dispositivos (como sensores y módulos) en un solo bus, permitiendo que un microcontrolador se comunique con varios periféricos de forma sencilla y eficiente. [24]

2.4 Marco teórico

Los siguientes trabajos investigativos son guías donde se detalla el proceso de estudio que apoya a la presente propuesta de titulación.

En el artículo titulado "A Wireless Sensor Network Air Pollution Monitoring System" escrito por Haibo Zhou y otros, se describe el diseño y la implementación de un sistema de monitoreo de la calidad del aire utilizando una red de sensores inalámbricos. Este sistema recopila datos de contaminantes del aire en tiempo real y transmite la información a una plataforma central para su análisis. La investigación destaca la importancia de los sensores IoT para proporcionar datos precisos y en tiempo real, lo que permite una mejor comprensión y gestión de la calidad del aire en diferentes áreas urbanas [25].

La tesis de Juan Carlos Mendoza, titulada "Desarrollo de un sistema de monitoreo meteorológico basado en IoT para la predicción de eventos climáticos extremos", aborda la utilización de sensores IoT para recopilar datos meteorológicos, los cuales son analizados mediante algoritmos de aprendizaje automático. Este trabajo tiene como objetivo principal incrementar la precisión de las predicciones meteorológicas mediante redes neuronales y técnicas avanzadas de análisis de datos. Asimismo, resalta las ventajas de integrar tecnologías IoT para obtener datos en tiempo real y aplicarlos en modelos predictivos.

Por su parte, la tesis de Laura García, titulada "Sistema de alerta temprana para eventos meteorológicos extremos utilizando tecnologías IoT y aprendizaje automático", se centra en el diseño de un sistema que combina sensores IoT con algoritmos de aprendizaje automático para anticipar eventos climáticos extremos como huracanes y tormentas. El sistema utiliza tanto datos históricos como en tiempo real para emitir alertas tempranas, facilitando una mejor preparación y respuesta ante desastres naturales. Este estudio evidencia la capacidad de las tecnologías IoT y el aprendizaje automático para optimizar los sistemas de alerta meteorológica.[27].

En el proyecto titulado "Implementación de una red de sensores IoT para la monitorización ambiental en zonas rurales" realizado por Miguel Ángel Rodríguez, detalla la creación de una red de sensores IoT diseñada para recoger datos ambientales en áreas rurales. El sistema mide variables como la temperatura, la humedad y la calidad del aire, transmitiendo esta

información a una plataforma central para su análisis. El trabajo enfatiza cómo la tecnología IoT puede mejorar el monitoreo ambiental y la gestión de recursos naturales en estas zonas[28].

CAPÍTULO III

3 DESARROLLO DE LA PROPUESTA

3.1 COMPONENTES DE LA PROPUESTA

La estación meteorológica se compone de elementos físicos y herramientas lógicas para la implementación y la obtención de datos. Entre los componentes físicos se encuentran el ESP32 que es el microcontrolador, sensores para la medición de temperatura, humedad, radiación UV y un anemómetro con veleta. Los componentes lógicos que son empleados en la estación son plataformas de software libre: Arduino IoT Cloud, IFTT para visualizar la información de los sensores.

3.1.1 Componentes físicos.

- **Sensor Termómetro e higrómetro**

El sensor esta protegido para tener resistencia al agua, nos permite conocer simultáneamente la temperatura y la humedad del entorno. Registra la temperatura, mientras que el higrómetro determina el nivel de humedad relativa en el aire, estos sensores son uno solo conectados en 3 cables, sus aplicaciones están dirigidas en sistemas de regulación climática, control y monitoreo.

Figura 11. Sensor Termómetro e higrómetro

Fuente: Autor

Especificaciones:

Tabla 2 Sensor Termómetro e higrómetro

Especificación	Descripción
Rango de temperatura interior	14 a 140 °F
Rango de temperatura exterior	-40 a 149 °F
Rango de humedad interior	10 a 99%
Rango de humedad exterior	10 a 99%
Precisión de temperatura	±1 °F (generalmente)
Precisión de humedad	±5% (generalmente)
Tipo de sensor	Sensor de temperatura y humedad capacitivo
Alimentación	Baterías (tipo de batería específico)
Funciones adicionales	Registro de máximo/mínimo, retroiluminación
Conectividad	Opcional (Bluetooth, Wi-Fi en modelos avanzados)

Fuente: [29]

- **Sensor Velocidad del viento**

Utilizado para medir la velocidad del viento. Funciona a través de diversos mecanismos, como hélices o copas, que giran cuando son impactados por el aire. La rapidez del giro se traduce en una medida de la velocidad del viento. Estos dispositivos son esenciales en meteorología, navegación y diversas aplicaciones industriales, ya que proporcionan datos valiosos sobre las condiciones del viento. Rango de velocidad del viento 0 a 100 mph.

Figura 12. Sensor velocidad del viento

Fuente: [30]

Tabla 3 Sensor de velocidad del viento

Especificación	Descripción
Rango de velocidad del viento	0 a 100 mph
Precisión	±1.0 mph (generalmente)
Tipo de sensor	Anemómetro
Salida	Digital (opcionalmente analógica)
Material	Plástico resistente a la intemperie
Conectividad	Wi-Fi

Fuente: [29]

- **Sensor dirección del viento Veleta**

El eje de la veleta no gira tan libremente como las copas de viento. Esto es así por diseño, la amortiguación que tiene evita que la veleta no gire con la mínima brisa, lo que genera un

viento variable. La resistencia adicional permite que la veleta cambia de 2 a 3 mph, lo que proporciona el seguimiento de la dirección del viento.

Figura 13. Sensor dirección del viento -Veleta

Fuente: [29]

Especificaciones:

Tabla 4 Sensor dirección del viento

Especificación	Descripción
Rango de dirección del viento	0 - 360º
Precisión	±3º (generalmente)
Tipo de sensor	Sensor de dirección (veleta)
Salida	Digital (opcionalmente analógica)
Material	Plástico resistente a la intemperie
Conectividad	Wi-Fi

Fuente: [29]

- **Sensor UV**

Este sensor proporciona datos en tiempo real sobre la intensidad de la radiación UV, que es importante para evaluar el riesgo de exposición solar y su impacto en la salud, como el riesgo de quemaduras o daños a la piel. El sensor UV está diseñado para ofrecer mediciones precisas y es ideal para aplicaciones profesionales en el monitoreo meteorológico.

Figura 14. Sensor UV

Fuente: [31]

Especificaciones:

Tabla 5 Radiación UV

Especificación	Descripción
Rango de radiación solar (luz)	0 a 200,000 Lux
Precisión	±10% (generalmente)
Tipo de sensor	Fotodetector
Salida	Digital (opcionalmente analógica)
Material	Plástico resistente a la intemperie
Conectividad	Wi-Fi

Fuente: [29]

- **Modulo cargador de batería Litio**

El módulo TP4056 es un cargador de baterías de litio diseñado para cargar baterías de iones de litio de una celda de manera eficiente y segura. Es muy popular en proyectos electrónicos debido a su tamaño compacto, facilidad de uso y capacidad para integrarse en dispositivos portátiles. Este módulo incorpora un controlador de carga lineal TP4056 y un sistema de protección contra sobrecarga, sobre corriente y cortocircuitos, lo que garantiza la seguridad durante la carga.

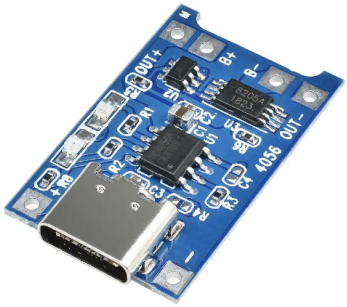


Figura 15 Modulo cargador de batería litio

Fuente: [33]

Tabla 6 Modulo cargador de batería

Especificación	Detalles
Voltaje de entrada (VIN)	4.5V a 5.5V
Voltaje de carga	4.2V ± 1%
Corriente de carga	Configurable hasta 1A (por defecto 1A)
Protección contra sobrecarga	Incluida
Protección contra sobredescarga	Incluida
Temperatura de operación	-10°C a +85°C
Indicadores LED	-**Rojo:**Cargando - **Azul:** Carga completa
Dimensiones	25 mm x 19 mm x 2 mm
Conexiones de entrada	Micro-USB o pines "IN+" y "IN-"
Conexiones de salida	Pines "BAT+" y "BAT-"

Fuente: [33]

- **Esp32 node mcu**

Este dispositivo cuenta con conectividad Wi-Fi y Bluetooth, lo que permite una fácil integración en redes inalámbricas para la recopilación y transmisión de datos. Diseñada para facilitar la creación de proyectos de Internet de las Cosas (IoT), lo que lo convierte en una excelente opción para proyectos que requieren comunicación en tiempo real y control de dispositivos de manera remota.

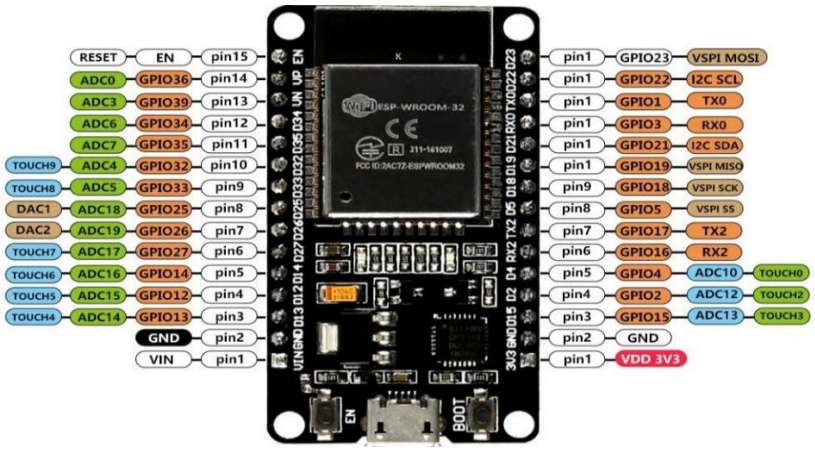


Figura 16. Microcontrolador ESP32

Fuente: [32]

Tabla 7 Especificaciones del microcontrolador ESP32

Especificación	**Descripción**
Modelo	ESP32 ESP-32 WiFi Bluetooth DevKit V1
Número de pines	30 pines
Microcontrolador	ESP32
Conectividad	WiFi 802.11 b/g/n, Bluetooth 4.2
Voltaje de alimentación	3.3V
Corriente de operación	Hasta 160 Ma
Temperatura de operación	-40 °C a 125 °C
Memoria Flash	4MB (generalmente)
GPIOs disponibles	34 GPIOs (algunos son compartidos)
Interfaz de comunicación	UART, SPI, I2C, PWM, ADC

Fuente: [32]

- **Panel Solar**

Este panel solar compacto y eficiente convierte la luz solar en energía eléctrica para alimentar dispositivos como el anemómetro, el higrómetro, el pluviómetro y otros sensores. Prolonga la vida útil de la batería y asegurando su funcionamiento sin interrupciones, generalmente opera a 2.5 a 3.6 voltios.

Figura 17. Panel solar

Fuente: Autor

Tabla 8 especificación del panel solar

Tipo de panel solar	**Panel solar de silicio cristalino**
Potencia nominal	5W a 20W (dependiendo del modelo)
Voltaje de salida	5V a 12V (generalmente)
Eficiencia	15-20%
Dimensiones	Variable (depende del modelo)
Peso	Variable (depende del modelo)
Conectividad	Conector MC4 (típicamente)
Resistencia a la intemperie	IP65 (resistente al agua)

Fuente: [29]

3.1.2 Componentes lógicos

- **Arduino IDE**

Diseñado para ser accesible tanto para principiantes como para usuarios avanzados, Arduino IDE utiliza un lenguaje de programación basado en C++ que es fácil de aprender, con funciones y bibliotecas integradas que simplifican el trabajo con sensores, motores, luces y otros componentes electrónicos. [33]

Los usuarios pueden desarrollar proyectos como sistemas de automatización, dispositivos de monitoreo, y proyectos de robótica, entre muchos otros. El IDE permite escribir código y cargarlo en un microcontrolador conectada a la computadora mediante un cable USB, lo cual permite que la placa funcione de forma autónoma una vez que el programa ha sido cargado.

- **Arduino IoT cloud**

A través de la Arduino IoT Cloud, los datos de sensores como temperatura, humedad, velocidad del viento y radiación UV pueden enviarse a la nube y visualizarse en dispositivos conectados, como teléfonos móviles o computadoras. Esta plataforma facilita la automatización de alertas, la creación de paneles personalizados lo que hace que el monitoreo meteorológico sea más accesible y flexible. [33]

- **IFTT**

Se puede configurar acciones como recibir notificaciones en tu teléfono si la temperatura alcanza un cierto nivel, o hacer que las luces de tu casa se apaguen cuando la estación detecta que está lloviendo. Al conectar la estación meteorológica con IFTTT, puedes integrar fácilmente la información meteorológica con otros dispositivos inteligentes y servicios en la nube, lo que hace que el monitoreo y la respuesta ante cambios climáticos sea más dinámico y personalizable. [34]

- **LSTM**

Esta red neuronal se aplica para analizar datos climáticos y predecir condiciones futuras. Utilizando Python, se entrena una red neuronal LSTM con datos históricos de temperatura, lo

que facilita la identificación de patrones complejos en la información. Esto permite realizar ajustes dinámicos en tiempo real, optimizando la respuesta ante cambios climáticos y proporcionando alertas más fiables. [35]

- **Thingspeak**

La información de los sensores puede ser visualizadas en gráficos interactivos y paneles personalizados, lo que facilita el análisis de tendencias a lo largo del tiempo. Además, ThingSpeak permite la integración con herramientas de análisis lo que mejora la capacidad de tomar decisiones basadas en datos meteorológicos precisos y actualizados. [36]

- **Python**

Python facilita la integración de diversos sensores de medición a través de librerías y módulos que permiten interactuar con hardware externo. Estos sensores envían datos en tiempo real que Python captura y procesa para obtener medidas precisas de las condiciones atmosféricas. Es sencillo guardar los datos meteorológicos en formatos como archivos CSV o bases de datos. Esto permite registrar de forma continua la información para realizar análisis históricos o estudiar patrones meteorológicos a largo plazo. [35]

Gracias a bibliotecas como Matplotlib, Pandas y NumPy, Python permite analizar los datos recopilados, generando gráficos y modelos de tendencia para interpretar la variabilidad en las condiciones atmosféricas.

3.2 DISEÑO DE LA PROPUESTA

Para el proyecto se desarrolla una estación portátil para monitorear las variables meteorológicas con tecnología IoT, se escogió los sensores adecuados para poder medir las condiciones climáticas que mas afecten al bienestar de los estudiantes como lo es la temperatura, humedad, radiación UV y velocidad del viento. Dado que estos parámetros son importantes para la predicción meteorológica se usa un sistema que registre la adquisición de datos.

Se procede a configurar el hardware y software para la adquisición de datos, utilizando la plataforma Arduino IoT Cloud, en este caso empleando un ESP32. Se desarrolla un enlace mediante IFTTT. Se desarrolla un flujo de datos sirve de base para el sistema de predicción, que utilizará redes neuronales LSTM en Python. Estos modelos se entrenarán utilizando los datos almacenados, y una vez entrenados, podrán predecir las condiciones meteorológicas futuras.

El microcontrolador por medio de comunicación serial realiza capturas de los datos recopilados por cada sensor para poder dividir su desempeño en el entrenamiento del modelo de redes neuronales LSTM. Posteriormente, se lleva a cabo un procesamiento de las señales climáticas para garantizar la precisión de los datos.

Una vez finalizado el análisis y el procesamiento de las señales, se procede a configurar el entorno en Spyder Python para implementar el modelo de la red LSTM. Esta etapa incluye la definición de parámetros clave, como el número de capas LSTM, la cantidad de neuronas por capa, las funciones de activación, los algoritmos de optimización, para su comprobación la matriz de correlación. La configuración de la red se ajusta específicamente para cada variable meteorológica (sensor), con el propósito de adaptar el modelo.

3.2.1 Implementación del sistema de adquisición de datos de la estación meteorológica

Los sensores que seleccionaron por sus características y desempeño, nos dirigimos a diseñar el sistema electrónico para la adquisición de datos a la nube.

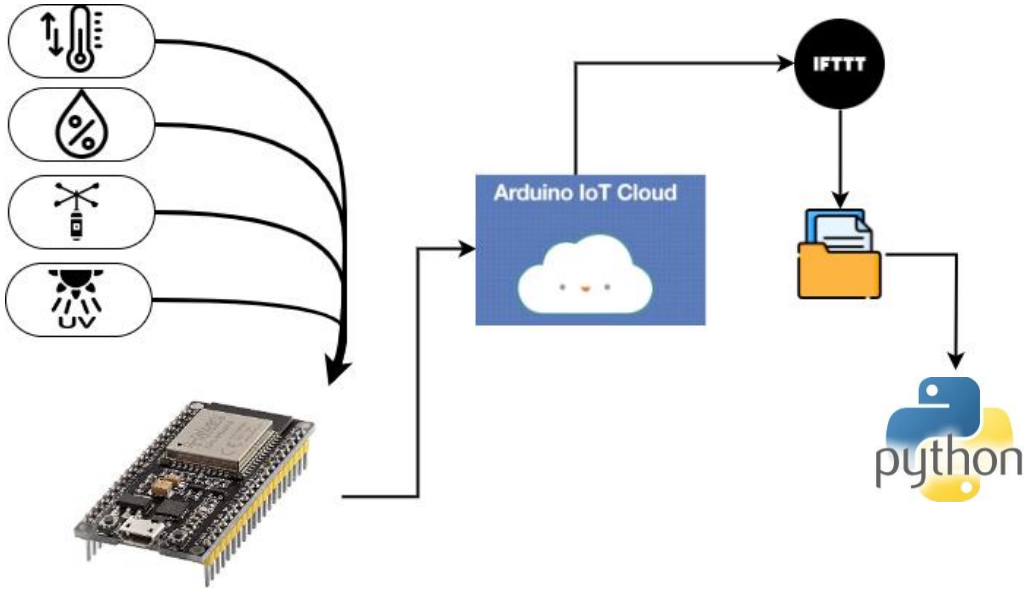


Figura 18. Conexiones físicas y lógicas

Fuente: Autor

La figura muestra las conexiones que se establecen al usar el ESP32 y la comunicación. A su vez conectándolos con los respectivos sensores. Para lograr una configuración completa que nos permita leer, medir y monitorear los datos meteorológicos.

El ESP32 se conecta mediante el protocolo MQTT hacia la plataforma Arduino IoT cloud, nos habilita la captura de datos del clima. Se establecen conexiones en IFTT para su análisis. Estos datos se pueden procesar en Matlab. Para obtener la red neuronal respectiva.

3.2.2 Configuración de elementos lógicos.

El siguiente diagrama de flujo presenta la obtención de datos de los sensores hacia la carpeta realizada para el prototipo.

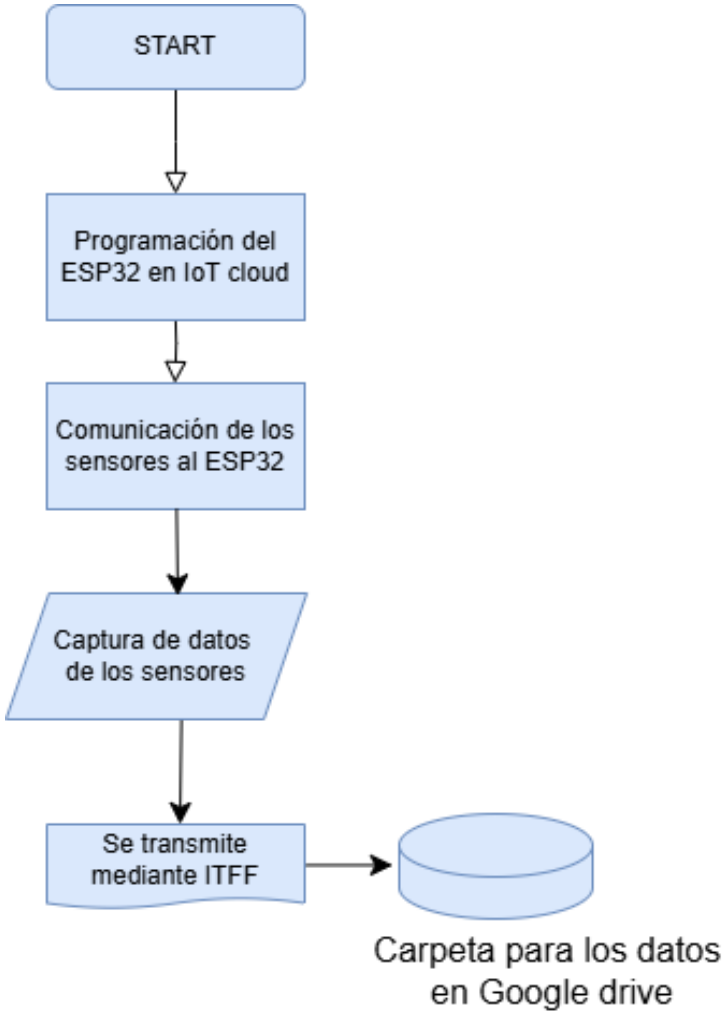


Figura 19. Flujo grama de la obtención de datos a drive

Fuente: autor

Repetimos el procedimiento para cada sensor, según la variable que se requiera medir. La estación recopilara eficientemente los datos requeridos, lo que nos lleva a tener un análisis de cada sensor en áreas específicas.

- **Programación del Esp32**

Las variables para medir se crean en Arduino IoT cloud en la pestaña THINGS, se usa el modelo NodeMCU-32s.

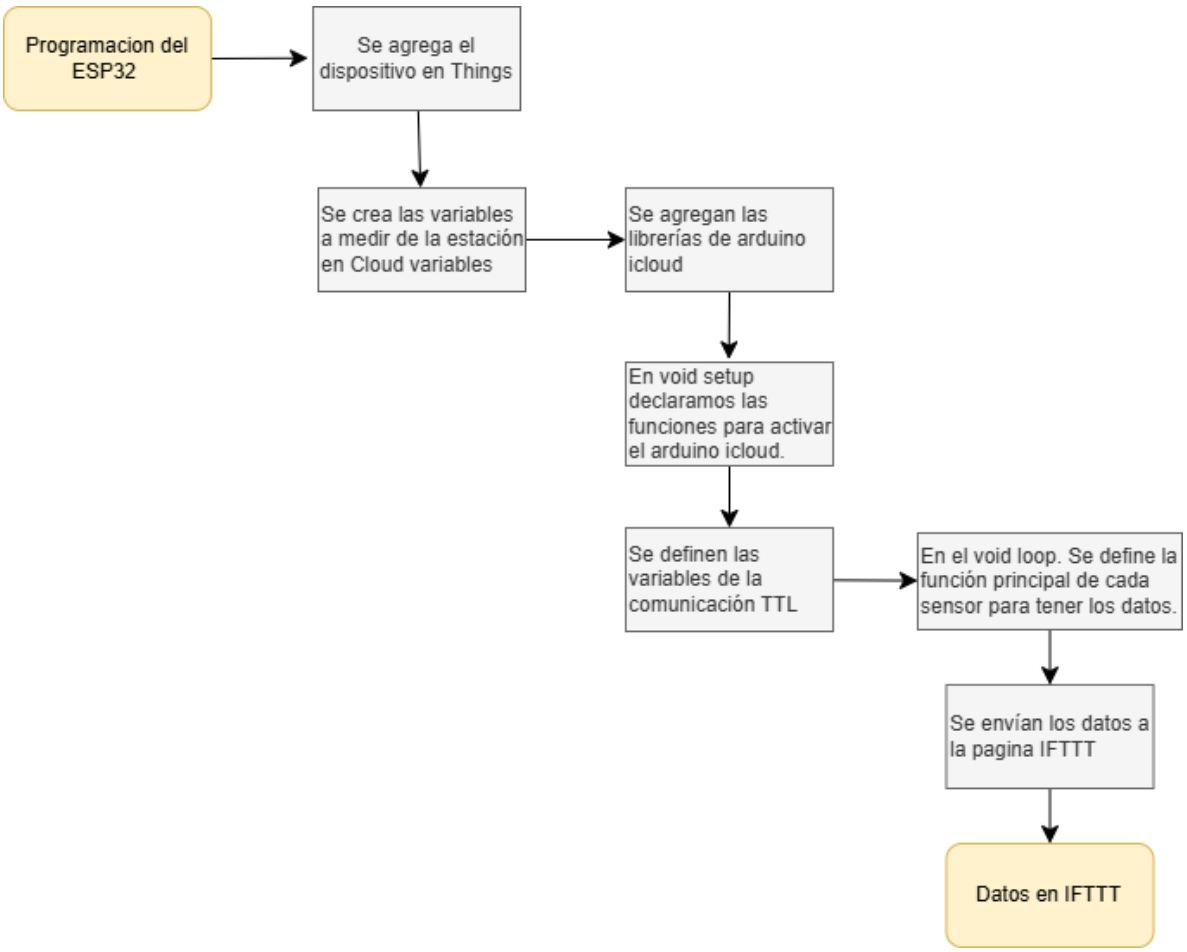


Figura 20. Programación del Esp32 en Arduino cloud

Fuente: Autor

Se procede a definir las variables necesarias en la sección de 'Variables en la Nube'. Esta plataforma hace que la programación en la nube al generar automáticamente un sketch, lo que resulta muy útil para el desarrollo de la estación de monitoreo y alarma meteorológica portátil con tecnología IoT.

Se instalan las librerías sugeridas por la plataforma para asegurar compatibilidad, y comunicación las variables se definirán utilizando los mismos nombres que se asignaron en Arduino Cloud, facilitando la comunicación y comprensible. Obteniendo las mediciones de los sensores de temperatura, humedad, velocidad, dirección del viento, radiación UV, se emplearán las funciones específicas, creando un api que ayudara a su vez con la transmisión de datos. Cuando los datos sigan transmitiendo por medio de la API mientras exista conexión wifi podrán ser analizados y visualizados.

Este enfoque no solo optimiza el monitoreo, sino que también permite el acceso a información en tiempo real sobre las condiciones meteorológicas, mejorando la efectividad de la estación de monitoreo

- **Programación en IFTT**

En la plataforma IFTTT, se configurará un evento que se vinculará a una cuenta de Google.

Esta herramienta proporciona un enlace a la programación desarrollada, el cual será integrado en la sección de webhook dentro de la pestaña 'THINGS' de Arduino IoT Cloud. Gracias a esta conexión, se podrán generar de manera automática hojas de cálculo en Google Drive para almacenar los datos obtenidos por la estación meteorológica.

3.2.3 Montaje de estación meteorológica portátil

Se realizo la instalación de una estación meteorológica portátil, para recolectar datos ambientales precisos que se puede mover fácilmente para medir temperatura, humedad, viento y radiación UV. Estos datos son importantes para predecir y estudiar el clima. La

estación es de fácil transporte, lo cual es conveniente para poder usarse en diferentes lugares, es muy útil para estudios de campo donde es necesario la recopilación de datos.

Figura 21 montaje de estación meteorológica

Fuente: Autor

3.2.4 Datos obtenido en Arduino Cloud

Después de que la información es recopilada y organizada en el microcontrolador (ESP32), se están listos para ser enviados a Arduino Cloud.
Esta página web te deja guardar, controlar y ver datos desde lejos, siendo perfecta para proyectos de IoT (Internet de las Cosas). El código para conectarse a la nube se configura en Arduino IDE usando internet.

Arduino IoT Cloud es la mejor opción para este proyecto porque puede recibir y controlar al instante diferentes datos del clima. A continuación, te mostramos una tabla con los eventos y su formato JSON.

Suceso	Valor	Formato
Evento	{ "d": { "value": "temp: 22.5" } }	JSON
Evento	{ "d": { "value": "humedad: 65" } }	JSON
Evento	{ "d": { "value": "radiación: 0.48" } }	JSON
Evento	{ "d": { "value": "viento: 3.5" } }	JSON
Evento	{ "d": { "value": "dirección: 180" } }	JSON

Tabla 9 tabla con los eventos recibidos

Este formato concede una fácil interpretación de los datos, manteniendo un registro de cada suceso en tiempo real en la plataforma IoT.

3.2.5 Conexión electrónica

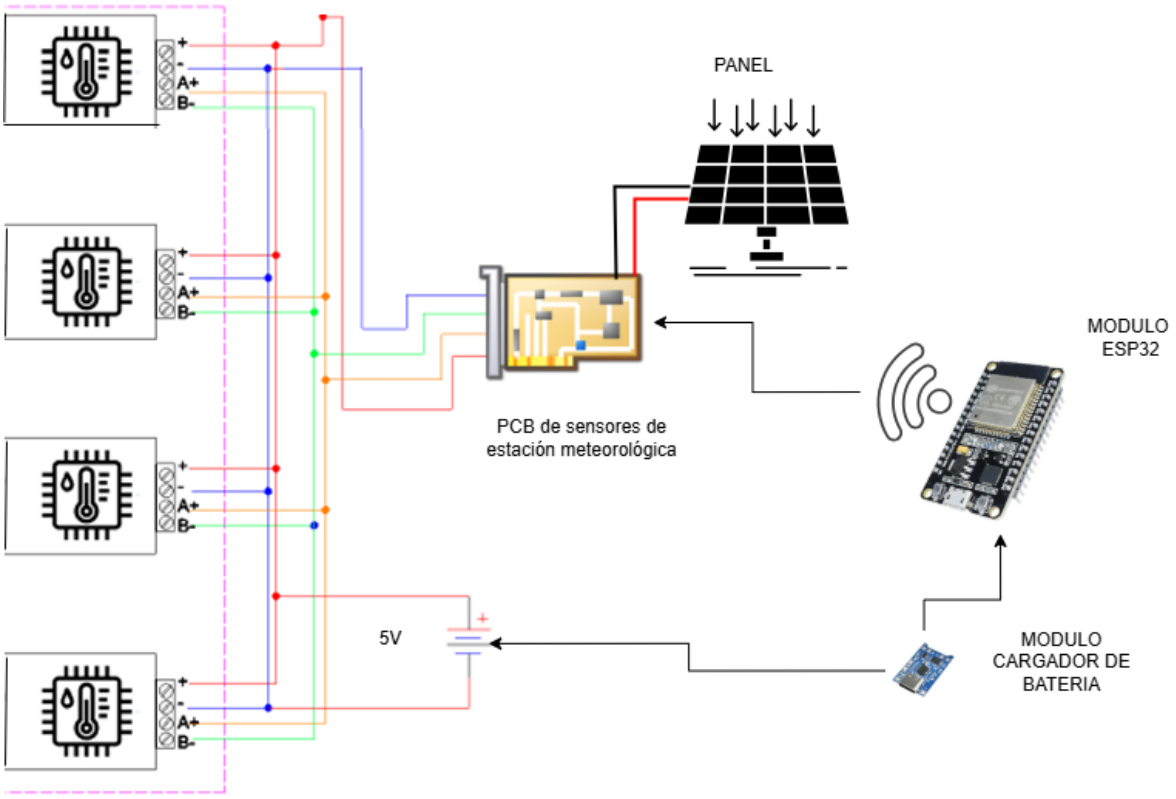
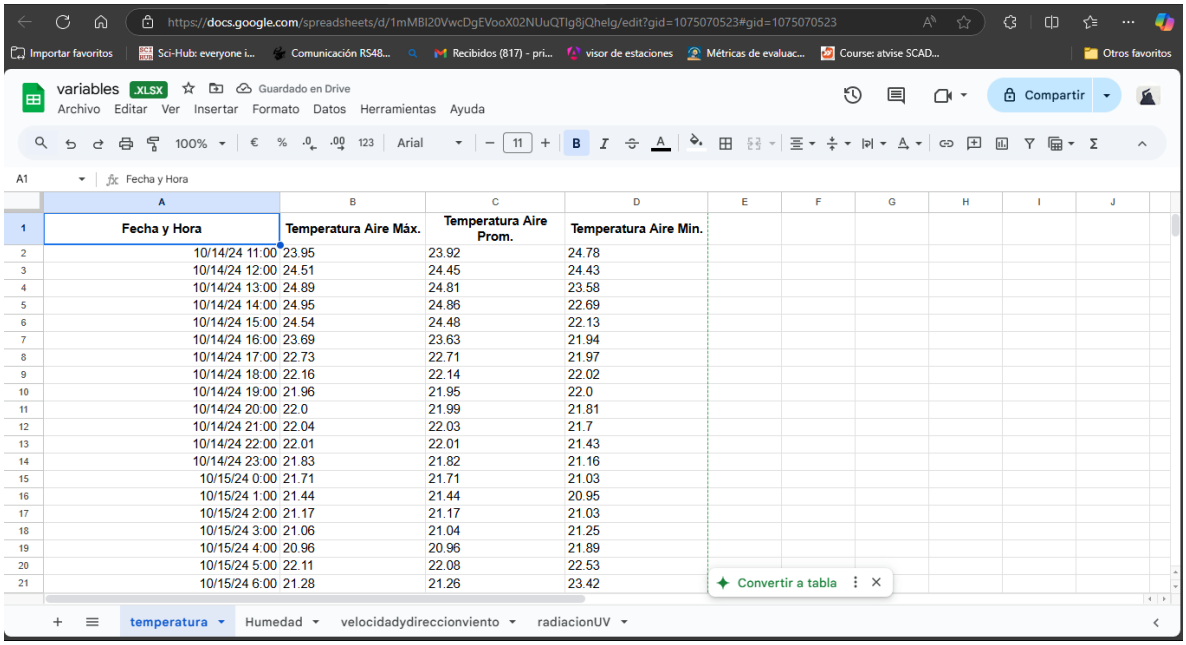


Figura 22 esquema electrónico

Fuente: Autor

3.2.6 Datos en Google drive

Cuando se configura un applet de IFTTT para grabar datos en Google Drive, los valores, como temperatura, humedad o velocidad del viento, se almacenan en una hoja de cálculo de Google Sheets en tiempo real o a intervalos específicos. Cada registro de datos generalmente incluye una marca de tiempo (fecha y hora) junto con las mediciones capturadas, lo que permite realizar un seguimiento detallado y organizado de la información. Al usar Google Drive, los datos están siempre disponibles en la nube, lo que permite acceder a ellos desde cualquier dispositivo con conexión a internet. Esto facilita el análisis y el monitoreo remoto.



variables .XLSX — Guardado en Drive

Archivo Editar Ver Insertar Formato Datos Herramientas Ayuda

Fecha y Hora	Temperatura Aire Máx.	Temperatura Aire Prom.	Temperatura Aire Min.
10/14/24 11:00	23.95	23.92	24.78
10/14/24 12:00	24.51	24.45	24.43
10/14/24 13:00	24.89	24.81	23.58
10/14/24 14:00	24.95	24.86	22.69
10/14/24 15:00	24.54	24.48	22.13
10/14/24 16:00	23.69	23.63	21.94
10/14/24 17:00	22.73	22.71	21.97
10/14/24 18:00	22.16	22.14	22.02
10/14/24 19:00	21.96	21.95	22.0
10/14/24 20:00	22.0	21.99	21.81
10/14/24 21:00	22.04	22.03	21.7
10/14/24 22:00	22.01	22.01	21.43
10/14/24 23:00	21.83	21.82	21.16
10/15/24 0:00	21.71	21.71	21.03
10/15/24 1:00	21.44	21.44	20.95
10/15/24 2:00	21.17	21.17	21.03
10/15/24 3:00	21.06	21.04	21.25
10/15/24 4:00	20.96	20.96	21.89
10/15/24 5:00	22.11	22.08	22.53
10/15/24 6:00	21.28	21.26	23.42

Convertir a tabla

temperatura | Humedad | velocidadydireccionviento | radiacionUV

Figura 23 datos grabados en google drive

Fuente: Autor

3.2.7 Configuración de la red neuronal

- **Fecha y Hora**: El aspecto temporal es esencial, ya que las condiciones climáticas varían considerablemente durante el día y entre diferentes estaciones del año.
- **Temperatura**: La temperatura registrada en el pasado nos ayuda a ver cuándo hace calor y cuándo frío, información quelas redes neuronales pueden usar para hacer mejores predicciones.

- **Humedad**: La humedad relativa nos dice cuánta humedad hay en el aire y este dato es importante para saber cómo se siente la temperatura y si va a llover o no. Además, está relacionada con la cantidad de nubes que hay en el cielo.
- **Radiación Solar**: La cantidad de rayos del sol que llega a la tierra afecta mucho a la temperatura y al clima en general. Esto es muy importante para poder predecir cambios en el tiempo.
- **Velocidad y Dirección del Viento**: Estas medidas son muy importantes para estudiar cómo el viento influye en el clima de u na zona y nos ayuda a prever tormentas u otros fenómenos atmosféricos.

Según su arquitectura, tiene un grupo de compuertas para regular cuánto de información debe olvidarse, actualizarse y entregarse. Todas emplean una función sigmoide (regresión logística) con un rango de cero (0 = bloquear todo) a uno (1 = dejar pasar todo) que describe la cantidad de información a dejar pasar.

3.2.7.1 Compuerta de Olvido:

- **Función en Predicción**: Esta compuerta ayuda al modelo a eliminar información que ya no es importante, lo que hace que las predicciones sean más precisas. Por ejemplo, si la información anterior no coincide con el clima actual, la computadora ajusta la memoria para concentrarse en datos más útiles.

3.2.7.2 Compuerta de Entrada:

- **Función en Predicción**: Es responsable de agregar nueva información, como las mediciones recientes de temperatura y humedad, al modelo de memoria. Esto es muy importante, porque ayuda a la LSTM a saber que datos son mas importantes para predecir el clima a corto plazo y como influyen en las condiciones del tiempo.

3.2.7.3 Compuerta de Salida:

- **Función en Predicción**: Descubre qué parte de la memoria usamos para hacer la predicción final. Esto garantiza que el modelo solo use la información más importante para hacer predicciones sobre el clima.

3.3.8 Construcción del Modelo

El modelo basado en LSTM (Long Short-Term Memory) Fue creado para predecir el clima como la temperatura, humedad, viento y radiación UV usando datos pasados. Este tipo de estructura es perfecta para analizar series de datos en el tiempo, ya que puede encontrar patronesy relaciones en la información secuencial.
Esto es muy importante para predecir el clima, que se basa en cambios a lo largo del tiempo.

La información meteorológica, como la temperatura, la humedad, la velocidad del viento y la radiación UVM se recopilo y guardo en Google Drive. Cada conjunto de datos se organizó en hojas separadas con etiquetas de fecha y hora. En Python, juntamos los datos en un solo DataFrame para hacer una serie de tiempo sin interrupciones.

Los datos se prepararon antes para cambiar el formato de las marcas de tiempo y asegurarse de que todas las variables estuvieran sincronizadas. También, ajustamos los valores de cada variable para que estén en una escala de 0 a 1 usando MinMaxScaler.

Se definió un modelo de red LSTM en Python utilizando las bibliotecas para la respectiva recolección de datos, sincronización, entrenamiento capaz, optimizador, transformación y predicciones, se debe asegurar la compatibilidad ver Tabla 9.

El modelo se configuró con una capa LSTM para comprender las relaciones temporales en los datos, seguida de una capa Dropout para evitar el sobreajuste. Se agregó una capa espesa con solo una célula nerviosa para hacer la predicción final. En la configuración del modelo, se establecieron ciertos valores importantes, como cuantos elementos puede tener se establecieron ciertos valores importantes, como ciertos valores importantes, como cuantos elementos puede tener la secuencia de entrada.

Tabla 10 Bibliotecas para la construcción del modelo LSTM

Biblioteca	Función Principal
NumPy	Gestión eficiente de matrices y operaciones matemáticas necesarias para procesar los datos.
Pandas	Lectura, manipulación y limpieza de datos provenientes de sensores almacenados en Google Drive.
TensorFlow/Keras	Entena y evalúa el entrenamiento LSTM,
scikit-learn	Normaliza y escala datos para mejorar el rendimiento del entrenamiento
Matplotlib	Visualiza creando gráficos para comparar entre los datos reales y la predicción.
OpenPyXL	Maneja archivos como Excel tipo .xlsx para poder llamar a los datos meteorológicos.

Fuente: Autor

La tabla mostrada ayuda a describir y justificar el uso de cada biblioteca.

- **Parámetros de la red configurados**

Tabla 11. Parámetros configurados.

Parámetro	Valor
Numero de capas LSTM	2
Unidades LSTM	50 en cada capa
Capa densa (oculta)	25 unidades

Capa de salida	1 unidad
Optimizador	Adam
Función de perdida	mean_squared_error
métrica	mean_squared_error
Épocas	50
Tamaño del lote	32

Fuente: Autor

- **Arquitectura de la red LSTM**

Tabla 12 Arquitectura de la red LSTM

Nombre de la Capa	Tipo de Capa	Forma de Entrada	Forma de Salida	Descripción
Entrada	Input	(n_samples, n_steps, n_features)	(n_samples, n_steps, n_features)	Ventana temporal con múltiples características.
LSTM-1	LSTM	(n_steps, n_features)	(n_steps, 50)	Primera capa LSTM que devuelve secuencias.
LSTM-2	LSTM	(n_steps, 50)	-50	Segunda capa LSTM que devuelve la última salida de la secuencia.
Dense-1	Dense (ReLU)	-50	-25	Capa densa oculta con activación ReLU. Introduce no linealidad.
Capa de salida	Dense (Lineal)	-25	-1	Capa densa para producir la predicción final escalar.

Fuente: Autor

Figura 24 Arquitectura de la red

Fuente: autor

3.3.8.1 Obtención del conjunto de datos

Este conjunto de variables representa los diferentes aspectos del flujo de trabajo en el modelo LSTM para la predicción de datos meteorológicos. Desde la carga y preparación de los datos (como se observa en los DataFrames df, humedad, radiacionUV), pasando por el escalado y preprocesamiento (data_scaled, padding), hasta la predicción y post-procesamiento (predictions, predictions_unscaled), todas estas variables colaboran para entrenar y evaluar el modelo en la predicción de variables climáticas.

Nombre	Tipo	Tamaño	Valor
data	DataFrame	[1180, 13]	Column names: Fecha y Hora, Temperatura Aire Máx., Temperatura Aire Pr ...
data_scaled	DataFrame	[1180, 13]	Column names: Temperatura Aire Máx., Temperatura Aire Prom., Temperatu ...
data_values	Array of float64	[1180, 12]	[[3.60218957e-16 0.00000000e+00 -3.61716889e-16 ... -1.31230024e-15 ...
df	DataFrame	[91, 4]	Column names: Fecha y Hora, Viento Max, Viento MIN, VIENTO
file_path	str	46	C:\Users\prisc\OneDrive\Escritorio\Libro1.xlsx
humedad	DataFrame	[261, 4]	Column names: Fecha y Hora, Humedad relativa Máx., Humedad relativa P ...
num_features	int	1	12
padding	Array of float64	[224, 11]	[[0. 0. 0. ... 0. 0. 0.] [0. 0. 0. ... 0. 0. 0.]
predicciones_nuevas	Array of float32	[224, 1]	[[0.00990522] [-0.03225509]
predictions	Array of float32	[224, 1]	[[0.00990522] [-0.03225509]
predictions_padded	Array of float64	[224, 12]	[[0.00990522 0. 0. ... 0. 0. 0. ...
predictions_unscaled	Array of float64	[224]	[56.75853391 55.92690899 55.45300018 ... 82.07810134 79.923889… 76.5 ...

Figura 25. datos exploradores de variables en Python

Fuente: autor

Nombre	Tipo	Tamaño	Valor
sequence_length	int	1	60
temperatura	DataFrame	[565, 4]	Column names: Fecha y Hora, Temperatura Aire Máx., Temperatura Aire Pr ...
train_size	int	1	896
velocidadydireccionviento	DataFrame	[91, 4]	Column names: Fecha y Hora, Viento Max, Viento MIN, VIENTO
X	Array of float64	[1120, 60, 11]	[[[3.60218957e-16 0.00000000e+00 -3.61716889e-16 ... -1.54567797e-16 ...
X_test	Array of float64	[224, 60, 11]	[[[1.54303600e+00 1.55220367e+00 1.56029802e+00 ... -6.44232851e-01 ...
X_train	Array of float64	[896, 60, 11]	[[[3.60218957e-16 0.00000000e+00 -3.61716889e-16 ... -1.54567797e-16 ...
y	Array of float64	[1120]	[-1.79987877 -1.8049484 -1.74664764 ... 0.44546086 0.457121… 0.5 ...
y_test	Array of float64	[224]	[3.60218957e-16 3.60218957e-16 3.60218957e-16 ... 4.45460861e-… 4.57 ...
y_test_padded	Array of float64	[224, 12]	[[3.60218957e-16 0.00000000e+00 0.00000000e+00 ... 0.00000000e… 0. ...
y_test_unscaled	Array of float64	[224]	[56.56315044 56.56315044 56.56315044 ... 65.35 65.58 68.13 ...
y_train	Array of float64	[896]	[-1.79987877e+00 -1.80494840e+00 -1.74664764e+00 ... 3.60218957e-16 . ...

Figura 26 datos exploradores de variables en Python

Fuente: Autor

El entorno accede a las variables y estructura de los datos usado durante la construcción y ejecución del modelo LSTM.

3.3.9 Matriz correlación de las variables

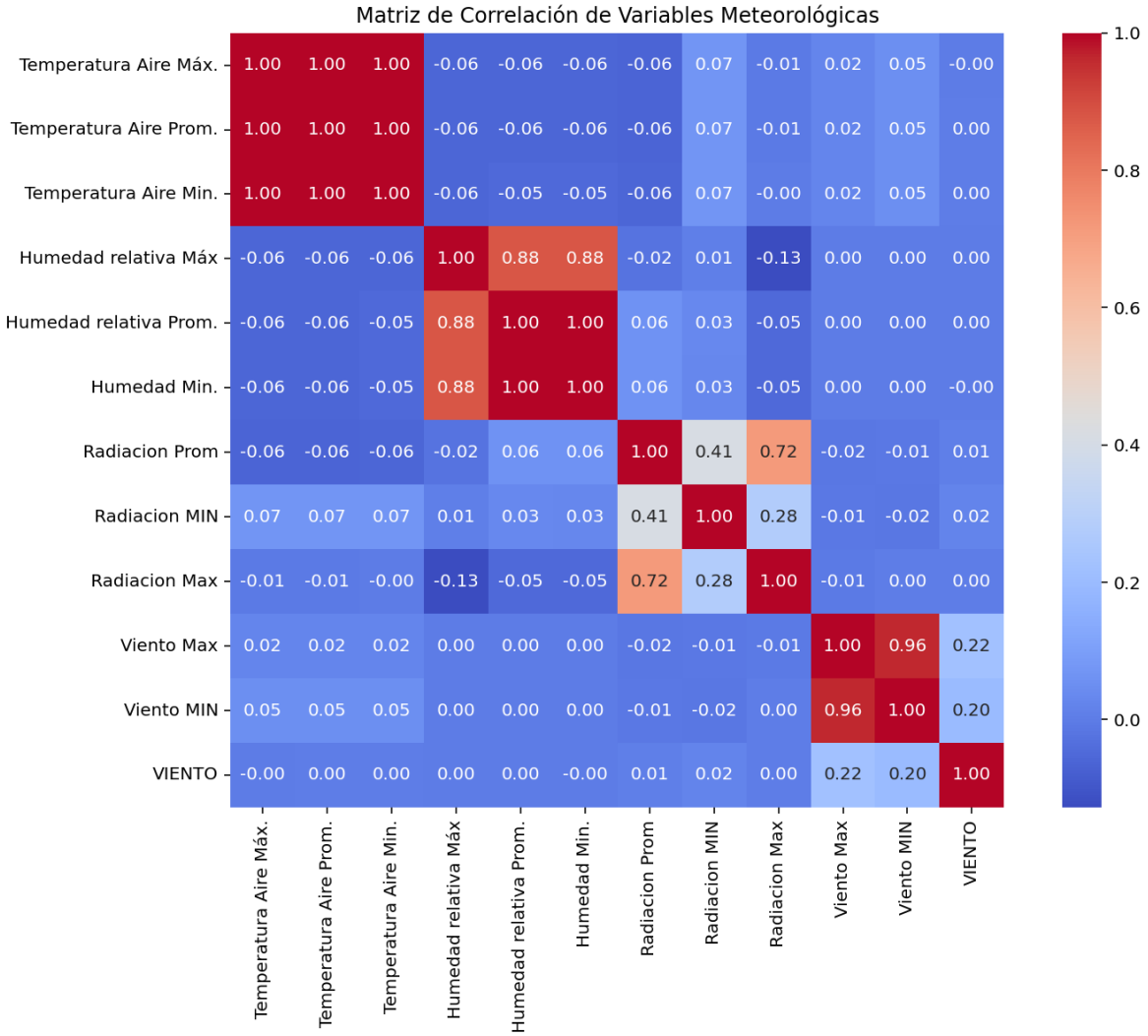


Figura 27 Matriz correlación de las variables meteorológicas

Fuente: Autor

La matriz de correlación de las variables adquiridas analiza cómo se relacionan entre sí las diferentes variables del conjunto de datos. Los valores de la matriz varían entre -1 y 1, donde:

- **1** indica una correlación positiva perfecta (ambas variables aumentan juntas).
- **-1** indica una correlación negativa perfecta (cuando una variable aumenta, la otra disminuye).
- **0** indica ausencia de correlación.

En este caso, se visualiza la matriz mediante una matriz para facilitar su interpretación.

Entre estas observaciones tenemos:

- Los valores máximos, promedios de la temperatura muestran relación entre ellas, a su vez índices de cambio significativos, las variables son útiles para poder realizar el modelo de predicción.
- Las variables de humedad relativa máxima, promedio y mínima también muestran una correlación elevada, lo cual es coherente con su naturaleza.
- La radiación solar máxima tiene relación moderada con el valor promedio de radiación, indicando que su asociación permite lograr una mejor predicción.
- La velocidad del viento máxima, mínima y promedio, brinda una conexión mostrando un patrón consistente en estas mediciones.

3.3.10 Entrenamiento del modelo de percepción

Primero, definimos la estructura del modelo, en este caso una red LSTM. La LSTM es especialmente útil aquí porque puede "recordar" patrones a largo plazo en los datos, lo cual es importante en series temporales como las condiciones meteorológicas. En cada paso de entrenamiento, el modelo procesa un lote de datos (mini-batch) desde la capa de entrada hasta la capa de salida.

El entrenamiento que logra la predicción compara el valor real usado con el de la función de perdida en este caso el Error Cuadrático Medio (MSE). La función va a evaluar la diferencia entre los valores reales y la predicción, minimizado el error. El modelo se va ajustando según sus conexiones internas este ajuste es hecho mediante un proceso conocido para implicar calcular la función de perdida respecto a cada variable del modelo.

El entrenamiento se realiza en múltiples épocas. En cada época, el modelo pasa por todo el conjunto de datos de entrenamiento y ajusta los pesos en función de los errores acumulados. Durante el entrenamiento, usamos un conjunto de validación (una porción del conjunto de datos no usada en el entrenamiento directo) para evaluar el rendimiento del modelo en datos no vistos. Esto permite identificar si el modelo está sobre ajustándose (overfitting) o si

necesita más ajustes. La pérdida de validación (val_loss) es observada en cada época para ver cómo se desempeña el modelo.

El proceso una vez culminado el entrenamiento, se prueba el modelo utilizando un conjunto de datos independiente para verificar su capacidad de muestreo y predicción. En este caso, se empleó el RMSE (Raíz del Error Cuadrático Medio) como métrica principal para medir la precisión de las predicciones relacionadas con las variables climáticas.

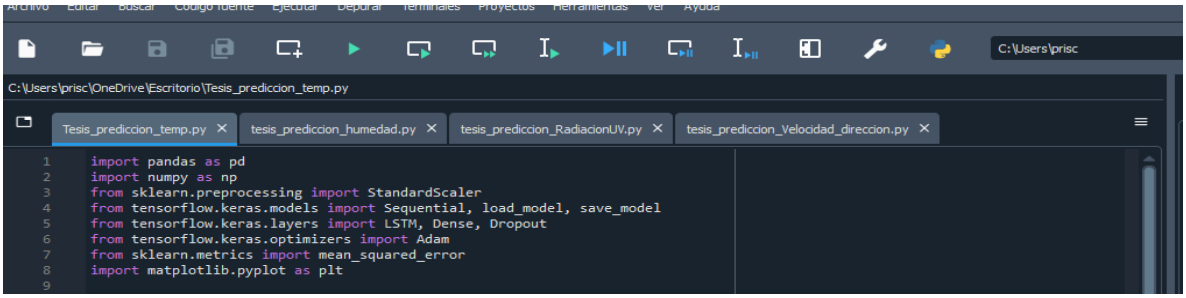


Figura 28 entrenamiento del modelo

Fuente: Autor

3.2.11 Resultados de las variables de la estación meteorológica

Temperatura: Los datos tomados durante una semana de recolección de datos, se muestra que solo en algunos días se identificaron patrones que confirmaban temperaturas elevadas entre ratos u horas. La gráfica de temperatura muestra un comportamiento típico de fluctuaciones diarias, con variaciones destacadas en los valores máximos y mínimos, especialmente en las fechas del 12 y 14 de noviembre. Se ve una combinación de días cálidos y probablemente soleados, alternados con días de temperaturas más moderadas.

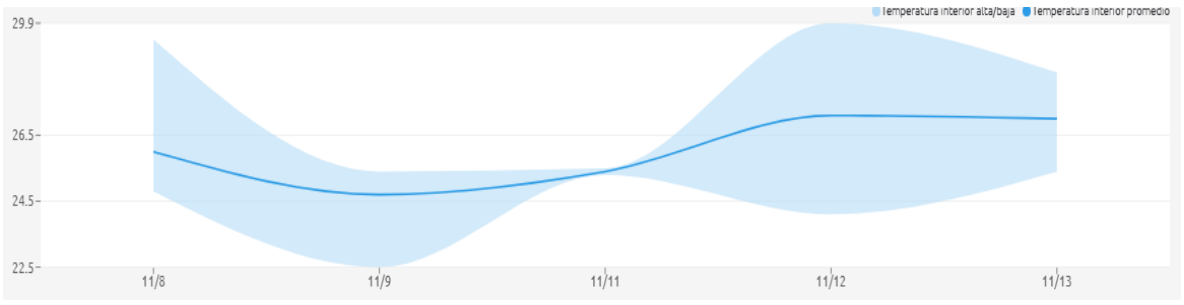


Figura 29 datos de la temperatura

Fuente: Autor

Velocidad del viento: La velocidad del viento es baja cuando la estación está dentro de un área cerrada, si se encuentra en zona cerrada no mostrara cambios bruscos ni mucho menos movimientos anormales indicando que dentro del espacio interior hay picos pequeños de velocidad.

Figura 30 datos de la velocidad del viento

Fuente: Autor

Humedad: Se muestra un clima mayormente húmedo y estable a lo largo de la semana, con una ligera disminución en los niveles de humedad, seguida de pequeñas variaciones en los días finales de igual manera son datos que son previstos tomados de zona interior la razón por la que no detecta cambios tan bruscos y se muestra estable, además que provenimos de un clima seco en la costa.

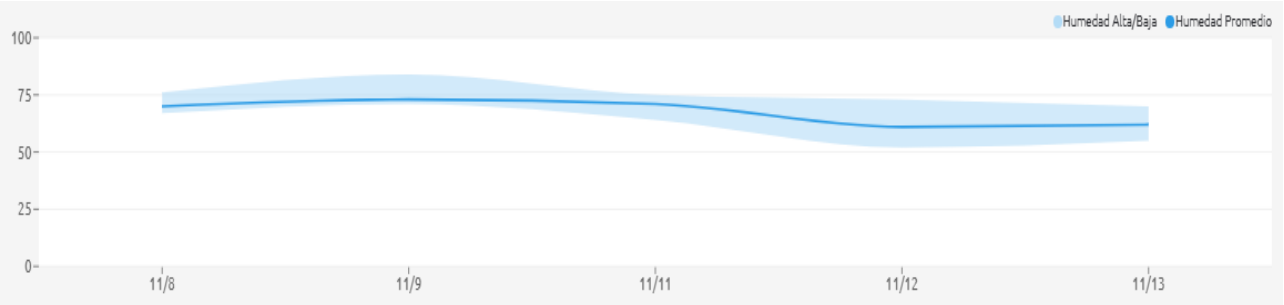


Figura 31 Datos de la humedad

Fuente: Autor

Radiación UV: La mayoría de los días (del 9 al 14 de noviembre), el índice de radiación UV es prácticamente nulo. Esto sugiere que durante estos días no hubo una exposición significativa a la radiación solar directa, lo cual puede ser indicativo de condiciones nubladas, lluviosas o de poca luz solar directa. Se observa un pico muy notable alrededor del 13 de noviembre, donde el índice de radiación UV alcanza un valor cercano a 3.

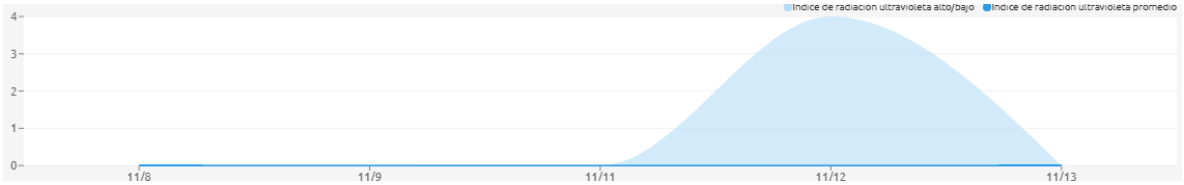


Figura 32 datos de la Radiación UV

Fuente: Autor

3.2.12 Pruebas y resultados de la predicción en Python

Predicción temperatura

El grafico representa la predicción de la temperatura en datos máximos, y la comparación con los valores reales. Se puede notar las variaciones de la temperatura que predice el modelo de manera variante, lo que nos indica que tienen una correlación un buen desempeño en la temperatura dado el modelo aplicado. Cuando la temperatura no permanece dentro de un rango el modelo intentar seguir de igual manera con exactitud, evidenciando las situaciones.

.

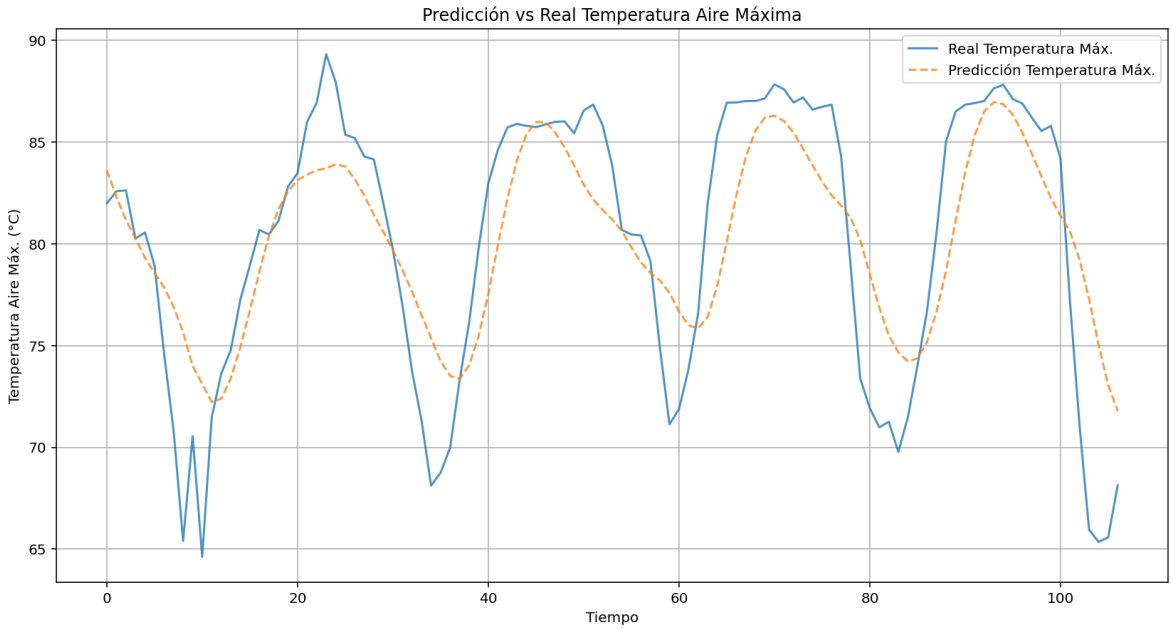


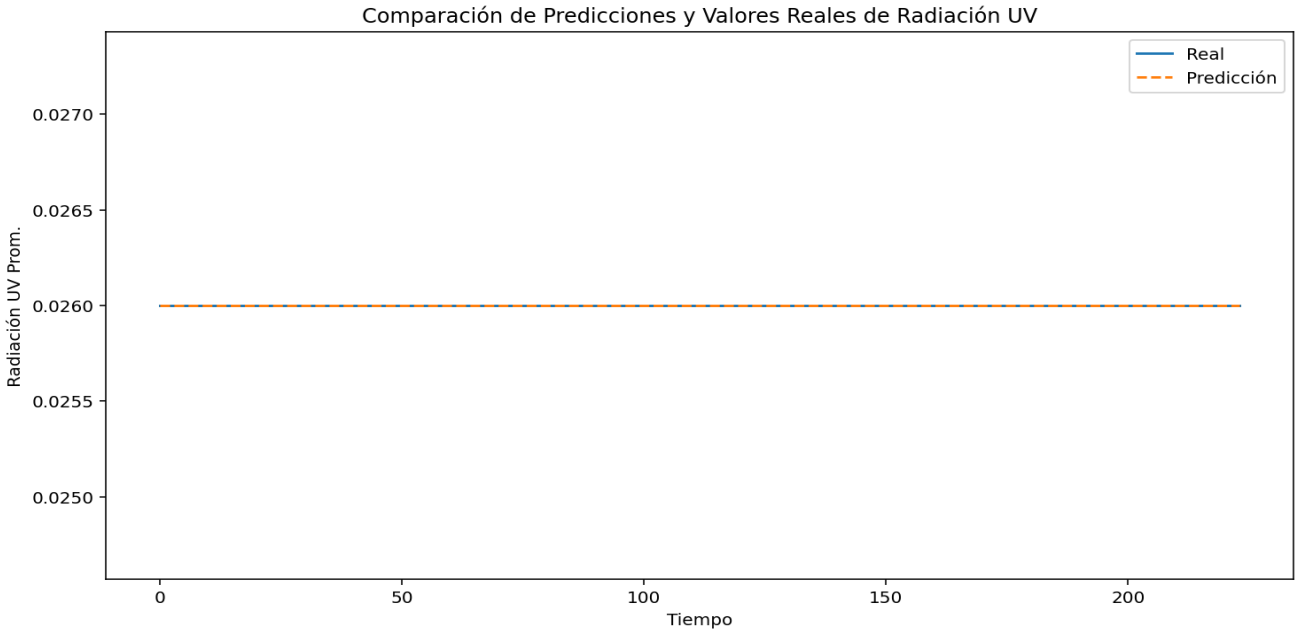
Figura 33 Predicciones y Valores Reales de Temperatura

Fuente: Autor

Radiacion UV

La radiación UV en la costa ha tenido tanto como valores elevados como bajos, sin embargo nuestro sensor nos muestra los datos tomados como elevaciones menores, y constantes excepto en las noches que nos encontramos en 0 sin embargo el modelo sigue las sugerencias de los datos tomados.

Esto puede deberse a que los datos de radiación UV recolectados durante el periodo registrado fueron muy consistentes.Segun los datos no se registra condiciones de mucha variabilidad.



Fuente: Autor

Figura 34Comparación de Predicciones y Valores Reales de Radiación UV

Velocidad del viento: Teniendo la gráfica muestra que el modelo reproduce estos valores de manera precisa en condiciones estables primero evaluamos los datos desde un espacio exterior donde ocurren diferentes tipos de cambios de velocidad ya sea bruscos o mínimos que muestran que existe picos el modelo intenta seguir los picos de igual forma hasta el momento en donde no haya movimiento dado los valores tomados.

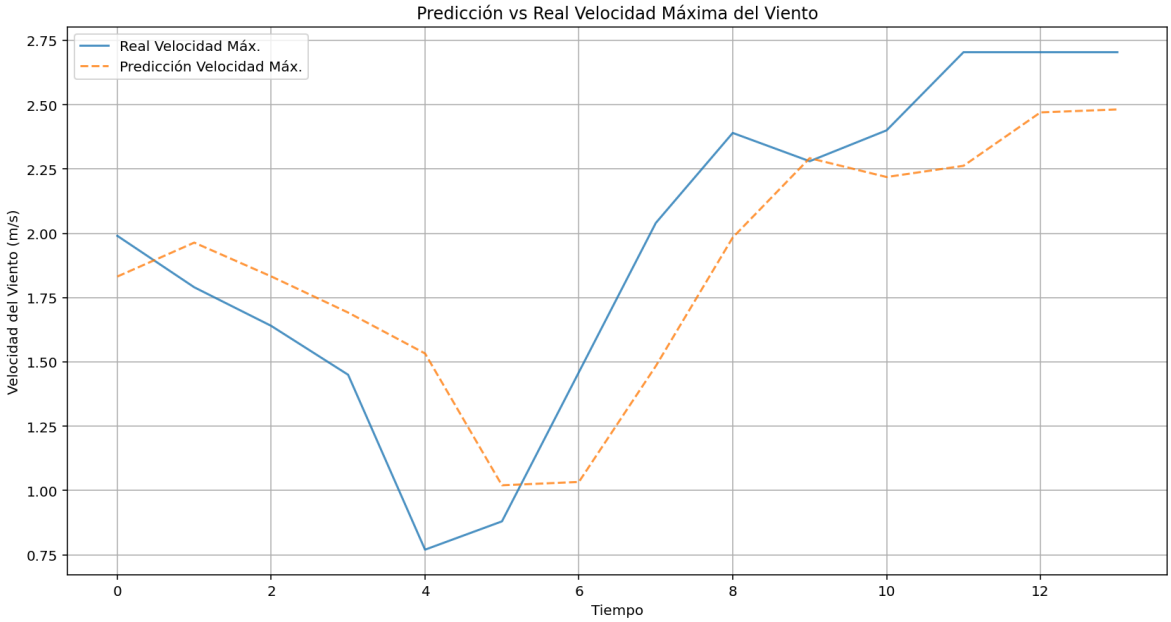


Figura 35 Comparación de Predicciones y Valores Reales de Velocidad del Viento

Fuente: Autor

Humedad: Esta constancia sugiere que, durante el periodo de análisis, el ambiente fue consistentemente húmedo, con niveles de humedad cercanos al 80%, sin cambios importantes en las condiciones atmosféricas que afectaran esta variable. la gráfica muestra que el modelo predice con precisión los valores de humedad relativa en condiciones estables.

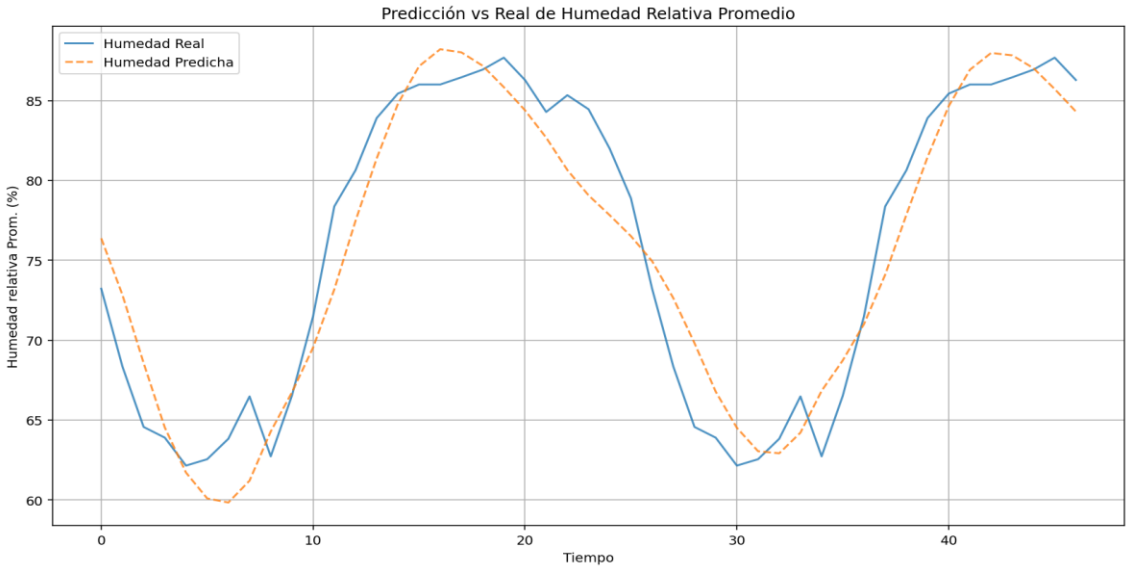


Figura 36 Comparación de Predicciones y Valores Reales de Humedad

Fuente: Autor

3.3.12 METRICAS

3.3.12.1 Métricas del modelo de aprendizaje

Se crea una tabla con los valores reales, predichos, diferencias absolutas y errores cuadrados.

El **RMSE** (Root Mean Squared Error) es un criterio usado para medir el error promedio de las predicciones, caracterizado por penalizar errores grandes. Se presenta en las mismas unidades que la variable objetivo, lo que permite interpretarlo fácilmente en el contexto del problema.

El **MAE** (Mean Absolute Error) analiza el promedio de las diferencias absolutas entre el valor real y las predicciones. Aporta una visión general de la magnitud del error sin dar una importancia excesiva a errores particularmente grandes.

El **R^2** (Coeficiente de Determinación) refleja el porcentaje de variación en los datos reales explicado por el modelo. Un valor cercano a 1 indica que el modelo tiene un buen desempeño, mientras que valores bajos señalan una capacidad limitada para capturar la relación entre las variables.

```
In [42]: %runfile 'C:/Users/prisc/OneDrive/Escritorio/ENTRENAMIENTO TESIS-LIS/Sin título4.py' --wdir
RMSE: 1.14
MAE: 1.10
R²: 0.92
          Valores Reales Valores Predichos Diferencia Absoluta Error Cuadrático
0                     80                79                   1                1
1                     82                81                   1                1
2                     85                84                   1                1
3                     90                88                   2                4
4                     87                86                   1                1
5                     85                84                   1                1
6                     83                82                   1                1
7                     81                80                   1                1
8                     78                77                   1                1
9                     76                75                   1                1
Métricas             ---               ---          MAE: 1.10       RMSE: 1.14
```

Figura 37Métrica de la variable temperatura

Fuente: Autor

```
In [43]: %runfile 'C:/Users/prisc/OneDrive/Escritorio/ENTRENAMIENTO TESIS-LIS/Sin título5.py' --wdir
RMSE (Humedad): 1.00
MAE (Humedad): 1.00
R² (Humedad): 0.89
         Valores Reales (Humedad)  ... Error Cuadrático
0                              60  ...                1
1                              62  ...                1
2                              64  ...                1
3                              63  ...                1
4                              65  ...                1
5                              66  ...                1
6                              67  ...                1
7                              68  ...                1
8                              69  ...                1
9                              70  ...                1
Métricas                      ---  ...       RMSE: 1.00

[11 rows x 4 columns]
```

Figura 38 Métrica de la variable Humedad

Fuente: Autor

```
In [47]: %runfile 'C:/Users/prisc/OneDrive/Escritorio/ENTRENAMIENTO TESIS-LIS/Sin
título6.py' --wdir
RMSE (Velocidad del Viento): 0.10 m/s
MAE (Velocidad del Viento): 0.10 m/s
R² (Velocidad del Viento): 0.95
MAPE (Velocidad del Viento): 2.37 %
MedAE (Velocidad del Viento): 0.10 m/s
Max Error (Velocidad del Viento): 0.10 m/s
         Valores Reales (m/s)  ... Error Cuadrático (m/s²)
0                         3.5  ...                    0.01
1                         4.0  ...                    0.01
2                         4.2  ...                    0.01
3                         3.8  ...                    0.01
4                         3.9  ...                    0.01
5                         4.5  ...                    0.01
6                         4.8  ...                    0.01
7                         5.0  ...                    0.01
8                         4.6  ...                    0.01
9                         4.3  ...                    0.01
Métricas                  ---  ...             RMSE: 0.10
```

Figura 39Métrica de la variable velocidad del viento

Fuente: Autor

```
In [48]: %runfile 'C:/Users/prisc/OneDrive/Escritorio/ENTRENAMIENTO TESIS-LIS/Sin título7.py' --wdir
RMSE (Radiación UV): 0.09 mW/cm²
MAE (Radiación UV): 0.09 mW/cm²
R² (Radiación UV): 0.95
MAPE (Radiación UV): 13.09 %
MedAE (Radiación UV): 0.10 mW/cm²
Max Error (Radiación UV): 0.10 mW/cm²
         Valores Reales (mW/cm²)  ... Error Cuadrático (mW/cm²²)
0                            0.2  ...                        0.01
1                            0.4  ...                         0.0
2                            0.6  ...                        0.01
3                            0.8  ...                        0.01
4                            1.0  ...                        0.01
5                            0.9  ...                        0.01
6                            1.1  ...                        0.01
7                            1.3  ...                        0.01
8                            1.5  ...                        0.01
9                            1.4  ...                        0.01
Métricas                                                RMSE: 0.09
```

Figura 40 Métrica de la variable Radiación UV

Fuente: Autor

CAPÍTULO IV

4 CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones

El sistema de comunicación diseñado con acceso remoto a la nube permitió monitorear variables físicas climáticas, y realizar análisis predictivo mediante inteligencia artificial, logrando mejorar la capacidad de anticipación y respuesta ante eventos meteorológicos.

Se ha logrado desarrollar el modelo de entrenamiento de predicción LSTM y la tecnología IoT para predecir las variables climatológicas de la universidad. La estación proporciono la información periódica de la temperatura, humedad, velocidad del viento y radiación UV. Además, se aplicaron técnicas para el entrenamiento, aplicando el MSE. Dado que estas permiten mostrar una predicción con bajo porcentaje de error, lo que resulta una combinación exitosa del ESP32 y los sensores, comunicándose al software libre, la eficiencia del monitoreo es demostrada dado los patrones que se escogieron o métricas que ayudaron a un mejor entrenamiento.

El análisis de la predicción de las variables meteorológicas que son tomadas en zonas externas como internas permitió diseñar este prototipo portátil pretendiéndose usar en cualquier ubicación. El estudio arrojo resultados exitosos lo que indica que los usuarios pueden comprender las variaciones de temperatura. Es posible lograr cambios significativos gracias a los mensajes que indican cambios tanto como temperatura y demás.

La metodología que ha sido adoptada dada la utilidad de la estación meteorológica indica que es una base fundamental para futuras aplicaciones en el campo de las condiciones meteorológicas.

4.2 Recomendaciones

Es de importancia tener en cuenta que la estación únicamente se limitó a la conexión y medición de 3 variables de suma importancia, pero es de conocer que en un campo o zona externa como lo es la universidad, hay muchas mas variables que pueden influir en un cambio climático por ello se recomienda la implementación de el prototipo como una parte de un sistema autónomo, pero con el objetivo de tener mejor implementación en el monitoreo.

Al evaluar los beneficios que se cuentan usando Arduino cloud a través de investigaciones. Los estudios incluyen la recopilación de datos meteorológicos durante meses en lo que la red neuronal no es entrenada y se puede comparar con los meses posteriores a la implementación.

ANEXOS

ESTRUCTURA FINAL DE LA TESIS

Anexo 1 programación ESP32-ArduinoCloud

```
#include <WiFi.h>

#include <WiFiClientSecure.h>

#include <ArduinoJson.h>

#include <ArduinoIoTCloud.h>

#include <Arduino_ConnectionHandler.h>



// Credenciales WiFi

const char* ssid = "LIS 8477";

const char* password = "12345678";



// Credenciales de Arduino Cloud

const char* DEVICE_LOGIN_NAME = "65530693-ff65-4918-ba47-86a8ef57a3d2";

const char* DEVICE_KEY = "BFalzVKa9XPuGp6ksqoJzNUld";



WiFiConnectionHandler ArduinoIoTPreferredConnection(ssid, password);



// Variables de Ambient Weather API

String apiKey = "afddf7216a7545c885479aee1ff2ad1e740c2166e00a47d295603adaa77f69ce";
```

```
String applicationKey =
"7ae602b5934c41ed984ccec522548e6f054db6611243473e92a6155faf8acdc9";
String deviceMac = "C8:C9:A3:27:A3:1A";


// Variables de Arduino Cloud
float Temperatura;
float Humedad;
float velocidaddelviento;
float direcciondeviento;


void setup() {
  Serial.begin(115200);
  delay(1500);


  // Configurar las propiedades en Arduino Cloud
  initProperties();
  ArduinoCloud.setBoardId(DEVICE_LOGIN_NAME);
  ArduinoCloud.setSecretDeviceKey(DEVICE_KEY);
  ArduinoCloud.begin(ArduinoIoTPreferredConnection);


  // Conectar a WiFi
  Serial.print("Conectando a WiFi...");
```

```
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("\nConectado a WiFi");

  Serial.println("Intentando conectarse a Arduino Cloud...");
}

void loop() {
  // Actualiza la conexión con Arduino Cloud
  ArduinoCloud.update();

  // Actualización de datos de Ambient Weather cada minuto
  static unsigned long lastUpdate = 0;
  if (millis() - lastUpdate > 60000) { // 60 segundos
    lastUpdate = millis();
    obtenerDatosAmbientWeather();
  }
}
```

```
void initProperties(){
  ArduinoCloud.addProperty(Temperatura, Permission::Read);
  ArduinoCloud.addProperty(Humedad, Permission::Read);
  ArduinoCloud.addProperty(velocidaddelviento, Permission::Read);
  ArduinoCloud.addProperty(direcciondeviento, Permission::Read);
}


void obtenerDatosAmbientWeather() {
  if (WiFi.status() == WL_CONNECTED) {
    WiFiClientSecure client;
    client.setInsecure();

    String url = "/v1/devices/" + deviceMac + "?apiKey=" + apiKey + "&applicationKey=" + applicationKey;

    if (client.connect("api.ambientweather.net", 443)) {
      client.print(String("GET ") + url + " HTTP/1.1\r\n" +
                   "Host: api.ambientweather.net\r\n" +
                   "Connection: close\r\n\r\n");

      // Ignorar encabezados HTTP
      while (client.connected()) {
```

```
  String line = client.readStringUntil('\n');
  if (line == "\r") {
    break;
  }
}

// Leer la respuesta HTTP completa para depuración
String response = "";
while (client.available()) {
  response += client.readString();
}

// Mostrar la respuesta completa del servidor
Serial.println("Respuesta completa del servidor:");
Serial.println(response);

// Verificar si la respuesta contiene datos JSON
int jsonStart = response.indexOf('[');
if (jsonStart != -1) {
  response = response.substring(jsonStart);
} else {
  Serial.println("No se encontró un array JSON en la respuesta.");
```

```
  return;
 }

 // Procesar el JSON
 DynamicJsonDocument doc(4096);
 DeserializationError error = deserializeJson(doc, response);

 if (error) {
  Serial.print("Error al parsear JSON: ");
  Serial.println(error.c_str());  // Mostrar error detallado
  return;
 }

 // Extraer datos del primer registro
 JsonObject firstEntry = doc[0];

 Temperatura = firstEntry["tempf"];             // Temperatura en °F
 Humedad = firstEntry["humidity"];              // Humedad en %
 velocidaddelviento = firstEntry["windspeedmph"]; // Velocidad del viento en mph
 direcciondeviento = firstEntry["winddir"];     // Dirección del viento en grados
```

```
      Serial.println("Datos actualizados en Arduino Cloud:");
      Serial.print("Temperatura: "); Serial.println(Temperatura);
      Serial.print("Humedad: "); Serial.println(Humedad);
      Serial.print("Velocidad del viento: "); Serial.println(velocidaddelviento);
      Serial.print("Dirección del viento: "); Serial.println(direcciondeviento);
    } else {
      Serial.println("Error al conectar con la API");
    }

    client.stop();
  } else {
    Serial.println("No conectado a WiFi");
  }
}
```

ANEXO 2 ENTRENAMIENTO TEMPERATURA

```
import pandas as pd

import numpy as np

from sklearn.preprocessing import MinMaxScaler

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import LSTM, Dense, Dropout

from tensorflow.keras.optimizers import Adam

from tensorflow.keras.callbacks import EarlyStopping

from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score

import matplotlib.pyplot as plt


# Cargar datos de la hoja de temperatura

file_path = r"C:\Users\prisc\OneDrive\Escritorio\Libro1.xlsx"

temperatura = pd.read_excel(file_path, sheet_name='temperatura')[['Fecha y Hora',
'Temperatura Aire Máx.']]


# Preprocesamiento

temperatura['Fecha y Hora'] = pd.to_datetime(temperatura['Fecha y Hora'])


# Reemplazar valores nulos

if temperatura['Temperatura Aire Máx.'].isnull().sum() > 0:
```

```
    temperatura['Temperatura Aire Máx.'] = temperatura['Temperatura Aire Máx.'].fillna(temperatura['Temperatura Aire Máx.'].mean())


# Normalizar los datos
scaler = MinMaxScaler(feature_range=(0, 1))
temperatura_scaled = scaler.fit_transform(temperatura[['Temperatura Aire Máx.']])


# Crear secuencias para el modelo LSTM
sequence_length = 30  # Tamaño de secuencia ajustado
def create_sequences(data, sequence_length):
    X, y = [], []
    for i in range(len(data) - sequence_length):
        X.append(data[i:i+sequence_length, 0])
        y.append(data[i + sequence_length, 0])
    return np.array(X), np.array(y)


X, y = create_sequences(temperatura_scaled, sequence_length)


# Dividir los datos en conjuntos de entrenamiento y prueba
train_size = int(len(X) * 0.8)
X_train, X_test = X[:train_size], X[train_size:]
y_train, y_test = y[:train_size], y[train_size:]
```

```
# Redimensionar para LSTM
X_train = X_train.reshape((X_train.shape[0], X_train.shape[1], 1))
X_test = X_test.reshape((X_test.shape[0], X_test.shape[1], 1))

# Definir el modelo LSTM mejorado
model = Sequential([
    LSTM(128, return_sequences=True, input_shape=(X_train.shape[1], 1)),
    Dropout(0.3),
    LSTM(64, return_sequences=False),
    Dropout(0.3),
    Dense(64, activation='relu'),
    Dense(32, activation='relu'),
    Dense(1)
])

# Compilar el modelo
model.compile(optimizer=Adam(learning_rate=0.0001),
loss='mean_squared_error')

# Entrenar el modelo
early_stopping = EarlyStopping(monitor='val_loss', patience=15,
restore_best_weights=True)
```

```
history = model.fit(
    X_train, y_train,
    epochs=300,
    batch_size=16,
    validation_data=(X_test, y_test),
    verbose=1,
    callbacks=[early_stopping]
)


# Realizar predicciones
predictions = model.predict(X_test)


# Desnormalizar las predicciones y los valores reales
predictions_unscaled = scaler.inverse_transform(predictions)
y_test_unscaled = scaler.inverse_transform(y_test.reshape(-1, 1))


# Evaluar el modelo
rmse = np.sqrt(mean_squared_error(y_test_unscaled, predictions_unscaled))
mae = mean_absolute_error(y_test_unscaled, predictions_unscaled)
r2 = r2_score(y_test_unscaled, predictions_unscaled)
print(f"RMSE: {rmse}")
print(f"MAE: {mae}")
```

```
print(f"R²: {r2}")

# Visualización de la pérdida durante el entrenamiento

plt.figure(figsize=(14, 7))

plt.plot(history.history['loss'], label="Pérdida de entrenamiento")

plt.plot(history.history['val_loss'], label="Pérdida de validación")

plt.title("Convergencia del modelo")

plt.xlabel("Épocas")

plt.ylabel("Pérdida")

plt.legend()

plt.grid()

plt.show()

# Visualizar los resultados

plt.figure(figsize=(14, 7))

plt.plot(y_test_unscaled, label="Real Temperatura Máx.", alpha=0.8)

plt.plot(predictions_unscaled, label="Predicción Temperatura Máx.", linestyle='--',
alpha=0.8)

plt.title("Predicción vs Real Temperatura Aire Máxima")

plt.xlabel("Tiempo")

plt.ylabel("Temperatura Aire Máx. (°C)")

plt.legend()

plt.grid()

plt.show()
```

ANEXO 3 ENTRENAMIENTO HUMEDAD

```
import pandas as pd

import numpy as np

from sklearn.preprocessing import MinMaxScaler

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import LSTM, Dense, Dropout

from tensorflow.keras.optimizers import RMSprop

from tensorflow.keras.callbacks import EarlyStopping

from sklearn.metrics import mean_squared_error, mean_absolute_error

import matplotlib.pyplot as plt


# Ruta al archivo

file_path = r"C:\Users\prisc\OneDrive\Escritorio\Libro1.xlsx"


# Cargar los datos de la hoja de Humedad

humedad = pd.read_excel(file_path, sheet_name='Humedad')[['Fecha y Hora', 'Humedad relativa Prom.']]

humedad['Fecha y Hora'] = pd.to_datetime(humedad['Fecha y Hora'])


# Reemplazar valores nulos

if humedad['Humedad relativa Prom.'].isnull().sum() > 0:
```

```
humedad['Humedad relativa Prom.'] = humedad['Humedad relativa Prom.'].fillna(humedad['Humedad relativa Prom.'].mean())


# Normalizar los datos
scaler = MinMaxScaler(feature_range=(0, 1))
humedad_scaled = scaler.fit_transform(humedad[['Humedad relativa Prom.']])


# Crear secuencias para el modelo LSTM
sequence_length = 30  # Ajustar longitud de secuencia
def create_sequences(data, sequence_length):
    X, y = [], []
    for i in range(len(data) - sequence_length):
        X.append(data[i:i+sequence_length, 0])
        y.append(data[i + sequence_length, 0])
    return np.array(X), np.array(y)


X, y = create_sequences(humedad_scaled, sequence_length)


# Dividir los datos en conjuntos de entrenamiento y prueba
train_size = int(len(X) * 0.8)
X_train, X_test = X[:train_size], X[train_size:]
y_train, y_test = y[:train_size], y[train_size:]
```

```
# Redimensionar para LSTM
X_train = X_train.reshape((X_train.shape[0], X_train.shape[1], 1))
X_test = X_test.reshape((X_test.shape[0], X_test.shape[1], 1))

# Definir el modelo LSTM mejorado
model = Sequential([
    LSTM(128, return_sequences=True, input_shape=(X_train.shape[1], 1)),
    Dropout(0.2),
    LSTM(64, return_sequences=False),
    Dropout(0.2),
    Dense(64, activation='relu'),
    Dense(32, activation='relu'),
    Dense(1)
])

# Compilar el modelo
model.compile(optimizer=RMSprop(learning_rate=0.0005), loss='mean_squared_error')

# Entrenar el modelo
early_stopping = EarlyStopping(monitor='val_loss', patience=15,
restore_best_weights=True)
```

```
history = model.fit(
    X_train, y_train,
    epochs=300,
    batch_size=16,
    validation_data=(X_test, y_test),
    verbose=1,
    callbacks=[early_stopping]
)


# Realizar predicciones
predictions = model.predict(X_test)


# Desnormalizar las predicciones y los valores reales
predictions_unscaled = scaler.inverse_transform(predictions)
y_test_unscaled = scaler.inverse_transform(y_test.reshape(-1, 1))


# Calcular métricas de evaluación
rmse = np.sqrt(mean_squared_error(y_test_unscaled, predictions_unscaled))
mae = mean_absolute_error(y_test_unscaled, predictions_unscaled)
print(f"RMSE: {rmse}")
print(f"MAE: {mae}")
```

```
# Visualización de la pérdida durante el entrenamiento

plt.figure(figsize=(14, 7))

plt.plot(history.history['loss'], label="Pérdida de entrenamiento")

plt.plot(history.history['val_loss'], label="Pérdida de validación")

plt.title("Convergencia del modelo")

plt.xlabel("Épocas")

plt.ylabel("Pérdida")

plt.legend()

plt.grid()

plt.show()


# Visualizar los resultados

plt.figure(figsize=(14, 7))

plt.plot(y_test_unscaled, label="Humedad Real", alpha=0.8)

plt.plot(predictions_unscaled, label="Humedad Predicha", linestyle='--', alpha=0.8)

plt.title("Predicción vs Real de Humedad Relativa Promedio")

plt.xlabel("Tiempo")

plt.ylabel("Humedad relativa Prom. (%)")

plt.legend()

plt.grid()

plt.show()
```

ANEXO 4 ENTRENAMIENTO VELOCIDAD DEL VIENTO

```
import pandas as pd

import numpy as np

from sklearn.preprocessing import MinMaxScaler

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import LSTM, Dense

from tensorflow.keras.optimizers import Adam

from tensorflow.keras.callbacks import EarlyStopping

import matplotlib.pyplot as plt


# Cargar los datos

file_path = "C:/Users/prisc/OneDrive/Escritorio/Libro1.xlsx"

data_vel_dir = pd.read_excel(file_path, sheet_name='velocidadydireccionviento')


# Verificar datos nulos

if data_vel_dir['Viento Max'].isnull().sum() > 0:

   data_vel_dir['Viento  Max'] = data_vel_dir['Viento  Max'].fillna(data_vel_dir['Viento Max'].mean())


# Visualizar los datos originales

plt.figure(figsize=(14, 7))

plt.plot(data_vel_dir['Viento Max'], label="Datos originales de Viento Max")
```

```
plt.title("Distribución de Viento Max")

plt.xlabel("Tiempo")

plt.ylabel("Velocidad del Viento (m/s)")

plt.legend()

plt.grid()

plt.show()


# Normalizar los datos con MinMaxScaler

scaler = MinMaxScaler(feature_range=(0, 1))

data_vel_max_scaled = scaler.fit_transform(data_vel_dir[['Viento Max']])


# Crear secuencias

def create_sequences(data, sequence_length):

    X, y = [], []

    for i in range(len(data) - sequence_length):

        X.append(data[i:i+sequence_length, 0])

        y.append(data[i + sequence_length, 0])

    return np.array(X), np.array(y)


sequence_length = 30

X, y = create_sequences(data_vel_max_scaled, sequence_length)
```

```
# Dividir los datos
train_size = int(len(X) * 0.8)
X_train, X_test = X[:train_size], X[train_size:]
y_train, y_test = y[:train_size], y[train_size:]

# Redimensionar X para LSTM
X_train = X_train.reshape((X_train.shape[0], X_train.shape[1], 1))
X_test = X_test.reshape((X_test.shape[0], X_test.shape[1], 1))

# Modelo básico para diagnóstico
model = Sequential([
    LSTM(50, input_shape=(sequence_length, 1), return_sequences=False),
    Dense(1)  # Una capa densa para la salida
])

model.compile(optimizer=Adam(learning_rate=0.001), loss='mean_squared_error')

# Entrenar el modelo
early_stopping = EarlyStopping(monitor='val_loss', patience=10, restore_best_weights=True)
history = model.fit(
    X_train, y_train,
```

```
    epochs=100,
    batch_size=16,
    validation_data=(X_test, y_test),
    verbose=1,
    callbacks=[early_stopping]
)
# Predicciones
predictions = model.predict(X_test)
predictions_unscaled = scaler.inverse_transform(predictions.reshape(-1, 1))
y_test_unscaled = scaler.inverse_transform(y_test.reshape(-1, 1))
# Visualizar resultados
plt.figure(figsize=(14, 7))
plt.plot(y_test_unscaled, label="Real Velocidad Máx.", alpha=0.8)
plt.plot(predictions_unscaled, label="Predicción Velocidad Máx.", linestyle='--', alpha=0.8)
plt.title("Predicción vs Real Velocidad Máxima del Viento")
plt.xlabel("Tiempo")
plt.ylabel("Velocidad del Viento (m/s)")
plt.legend()
plt.grid()
plt.show()
```

ANEXO 5 ENTRENAMIENTO RADIACIÓN UV

```
import pandas as pd

import numpy as np

from sklearn.preprocessing import StandardScaler

from tensorflow.keras.models import Sequential, load_model, save_model

from tensorflow.keras.layers import LSTM, Dense, Dropout

from tensorflow.keras.optimizers import Adam

from sklearn.metrics import mean_squared_error

import matplotlib.pyplot as plt


# Ruta al archivo

file_path = r"C:\Users\prisc\OneDrive\Escritorio\Libro1.xlsx"


# Cargar cada hoja con las columnas necesarias

temperatura = pd.read_excel(file_path, sheet_name='temperatura')[['Fecha y Hora', 'Temperatura Aire Máx.', 'Temperatura Aire Prom.', 'Temperatura Aire Min.']]

humedad = pd.read_excel(file_path, sheet_name='Humedad')[['Fecha y Hora', 'Humedad relativa  Máx.', 'Humedad relativa Prom.', 'Humedad Min.']]

radiacionUV = pd.read_excel(file_path, sheet_name='radiacion UV')[['Fecha y Hora', 'Radiacion Prom', 'Radiacion MIN', 'Radiacion Max']]

velocidadydireccionviento = pd.read_excel(file_path, sheet_name='velocidadydireccionviento')[['Fecha y Hora', 'Viento Max', 'Viento MIN', 'VIENTO']]
```

```
# Convertir la columna de fecha y hora a formato datetime
for df in [temperatura, humedad, radiacionUV, velocidadydireccionviento]:
    df['Fecha y Hora'] = pd.to_datetime(df['Fecha y Hora'])

# Unir las hojas usando outer join y llenar NaN temporalmente
data = pd.merge(temperatura, humedad, on='Fecha y Hora', how='outer')
data = pd.merge(data, radiacionUV, on='Fecha y Hora', how='outer')
data = pd.merge(data, velocidadydireccionviento, on='Fecha y Hora', how='outer')

# Reemplazar NaN e infinitos con el promedio de cada columna
data.replace([np.inf, -np.inf], np.nan, inplace=True)
data.fillna(data.mean(), inplace=True)

# Normalizar los datos usando StandardScaler, excluyendo la columna 'Fecha y Hora'
scaler = StandardScaler()
data_scaled = scaler.fit_transform(data.drop(columns=['Fecha y Hora']))

# Convertir a DataFrame y agregar 'Fecha y Hora' de vuelta
data_scaled = pd.DataFrame(data_scaled, columns=data.columns[1:])
data_scaled['Fecha y Hora'] = data['Fecha y Hora'].values
```

```
# Crear secuencias para el modelo LSTM con tamaño ajustado de secuencia
sequence_length = 60

def create_sequences(data, sequence_length, target_column_index):
    X, y = [], []
    for i in range(len(data) - sequence_length):
        X.append(data[i:i+sequence_length, :-1])  # Todas las columnas menos la última (Fecha y Hora)
        y.append(data[i + sequence_length, target_column_index])  # Predicción de la columna de radiación
    return np.array(X), np.array(y)

# Seleccionar el índice de la columna de "Radiacion Prom"
target_column_index = data.columns.get_loc("Radiacion Prom") - 1  # Restamos 1 por el desplazamiento de 'Fecha y Hora'

# Crear secuencias y dividir en conjuntos de entrenamiento y prueba
data_values = data_scaled.drop(columns=['Fecha y Hora']).values
X, y = create_sequences(data_values, sequence_length, target_column_index)

train_size = int(len(X) * 0.8)
X_train, X_test = X[:train_size], X[train_size:]
y_train, y_test = y[:train_size], y[train_size:]
```

```
# Definir un modelo LSTM simplificado
model = Sequential([
    LSTM(units=30, return_sequences=True, input_shape=(X_train.shape[1], X_train.shape[2])),
    Dropout(0.1),
    LSTM(units=15, return_sequences=False),
    Dropout(0.1),
    Dense(units=1)
])

# Compilar el modelo con una tasa de aprendizaje ajustada
model.compile(optimizer=Adam(learning_rate=0.0001), loss='mean_squared_error')

# Entrenar el modelo
model.fit(X_train, y_train, epochs=50, batch_size=32, validation_data=(X_test, y_test))

# Realizar predicciones en el conjunto de prueba
predictions = model.predict(X_test)

# Desnormalizar las predicciones y los valores de prueba
num_features = data_scaled.shape[1] - 1  # Excluyendo la columna 'Fecha y Hora'
```

```
padding = np.zeros((predictions.shape[0], num_features - 1))

predictions_padded = np.concatenate([predictions, padding], axis=1)

predictions_unscaled = scaler.inverse_transform(predictions_padded)[:, target_column_index]


y_test_padded = np.concatenate([y_test.reshape(-1, 1), padding], axis=1)

y_test_unscaled = scaler.inverse_transform(y_test_padded)[:, target_column_index]


# Evaluar el modelo

rmse = np.sqrt(mean_squared_error(y_test_unscaled, predictions_unscaled))

print("RMSE:", rmse)


# Visualización de resultados

plt.figure(figsize=(12, 6))

plt.plot(y_test_unscaled, label="Real")

plt.plot(predictions_unscaled, label="Predicción", linestyle='--')

plt.title("Comparación de Predicciones y Valores Reales de Radiación UV")

plt.xlabel("Tiempo")

plt.ylabel("Radiación UV Prom.")

plt.legend()

plt.show()
```

```
# Guardar el modelo entrenado

save_model(model, "modelo_lstm_radiacionUV_simplificado.h5")

print("Modelo guardado como 'modelo_lstm_radiacionUV_simplificado.h5'")


# Cargar el modelo guardado para futuras predicciones (opcional)

modelo_cargado = load_model("modelo_lstm_radiacionUV_simplificado.h5")

predicciones_nuevas = modelo_cargado.predict(X_test)
```

ANEXO 6 MATRIZ CORRELACIÓN

```
Import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt


# Ruta al archivo de datos

file_path = r"C:\Users\prisc\OneDrive\Escritorio\Libro1.xlsx"


# Cargar las hojas del archivo Excel

temperatura = pd.read_excel(file_path, sheet_name='temperatura')[['Fecha y Hora', 'Temperatura Aire Máx.', 'Temperatura Aire Prom.', 'Temperatura Aire Min.']]

humedad = pd.read_excel(file_path, sheet_name='Humedad')[['Fecha y Hora', 'Humedad relativa Máx', 'Humedad relativa Prom.', 'Humedad Min.']]
```

```
radiacionUV = pd.read_excel(file_path, sheet_name='radiacionUV')[['Fecha y Hora', 'Radiacion Prom', 'Radiacion MIN', 'Radiacion Max']]

velocidadydireccionviento = pd.read_excel(file_path, sheet_name='velocidadydireccionviento')[['Fecha y Hora', 'Viento Max', 'Viento MIN', 'VIENTO']]


# Convertir la columna 'Fecha y Hora' a formato datetime

for df in [temperatura, humedad, radiacionUV, velocidadydireccionviento]:

    df['Fecha y Hora'] = pd.to_datetime(df['Fecha y Hora'])


# Unir las hojas por la columna 'Fecha y Hora'

data = pd.merge(temperatura, humedad, on='Fecha y Hora', how='outer')

data = pd.merge(data, radiacionUV, on='Fecha y Hora', how='outer')

data = pd.merge(data, velocidadydireccionviento, on='Fecha y Hora', how='outer')


# Llenar valores faltantes con la media de cada columna

data.fillna(data.mean(), inplace=True)


# Calcular la matriz de correlación

correlation_matrix = data.drop(columns=['Fecha y Hora']).corr()


# Mostrar la matriz de correlación como un DataFrame

print("Matriz de Correlación:")
```

```
print(correlation_matrix)

# Visualizar la matriz de correlación como un heatmap

plt.figure(figsize=(12, 8))

sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f", cbar=True, square=True)

plt.title('Matriz de Correlación de Variables Meteorológicas')

plt.show()
```

Bibliografía

[1] O. D. L. N. UNIDAS, *Accion por el clima.*

[2] I. I. D. H. Y. M. Y. E. AMBIENTALES. [En línea]. Available: http://www.ideam.gov.co/documents/24277/889227/Anexo+T%C3%A9cnico+01/20a0adb6-9091-49b6-bada-1a4b962db907?version=1.0.

[3] I. J. O. M. I. A. A. P. E. Ing. Rolando Célleri PhD, «EVALUACIÓN DE LA VULNERABILIDAD Y RIESGO CLIMÁTICO DE LA GENERACIÓN DE HIDROENERGÍA POR LOS EFECTOS DE LA VARIABILIDAD Y CAMBIO CLIMÁTICO, ASÍ COMO LA IDENTIFICACIÓN DE MEDIDAS DE ADAPTACIÓN Y GENERACIÓN DE UN MECANISMO DE SEGUIMIENTO Y MONITOREO,» 31 AGOSTO 2020.

[4] IDEAM., «IDEAM- Diseño de la Red Hidrometeorológica Nacional, 1-16,» 2017. [En línea]. Available: http://sgi.ideam.gov.co/documents/412030/561097/M-GDI-H-G001+GU%C3%8DA+DISE%C3%91O+DE+LA+RED+HIDROMETEOROL%C3%93GICA+NACIONAL.pdf/9da0e118-58cc-43eb-87e0-8c6316dc691c?version=1.0#:~:text=El.

[5] E. L. R. N. Jorge Luis Ceballos Lievano, *Informe de Actividades Glaciologicas: Volcan nevado Santa Isabel-Sierra nevada,* 2008-2009.

[6] K. Puga, «INAMHI : RADIACION ULTRAVIOLETA,» *Inamhi,* 2023.

[7] C. EP, «CORPORACION ELECTRICA DEL ECUADOR- COCA CODO SINCLAIR,» 2023. [En línea]. Available:

https://www.celec.gob.ec/cocacodo/noticias/celec-ep-implementara-sistema-de-alertas-tempranas-en-la-central-coca-codo-sinclair/.

[8] H. P. C. Campoverde y C. C. U. Ávila, *"Análisis espacio-temporal meteorológico en una cuenca andina tropical del sur de Ecuador",* Cuenca, 2020.

[9] AEBE, «Red de Estaciones Meteorológicas Un camino hacia la sostenibilidad del sector bananero ecuatoriano.,» AGOSTO 2022. [En línea]. Available: https://www.aebe.com.ec/_files/ugd/f4cd67_1c68256846a746b282c945b68e6d9aa0.pdf?index=true.

[10] C. Campetella, B. Cerne y P. Salio, Entornos invisibles de la ciencia y la tecnologia (estacion metereologica), Buenos aires, 2011.

[11] B. P. H. D. J. M. A. ParedesMorillo George Joseth, *ESTACIÓN METEOROLÓGICA AUTOMÁTICA Y MEDICIÓN DE VARIABLES.*

[12] cne, «Capitulo II , Fenomenos atmosfericos y cambio climatico,» [En línea]. Available: https://www.cne.go.cr/CEDO-CRID/pdf/spa/doc231/doc231-2.pdf.

[13] C. Cortes, *EL INTERNET DE LAS COSAS: MAS INTERNET QUE OTRA COSA,* CENTRO DE ESTUDIOS EN LIBERTAD DE EGRESION Y ACCESO A LA INFORMACION .

[14] A. Lliso Cosin, *Uso de MQTT para el control de dispositivos de IoT,* Valencia, 2021.

[15] V. Teotia, «naukri 360,» 26 Marzo 2024. [En línea]. Available: https://www.naukri.com/code360/library/aws-iot-core. [Último acceso: Mayo 2024].

[16] H. y. PaperMod, «Alvaro Viebrantz,» 2017. [En línea]. Available: https://aviebrantz.com/articles/2017/2017-10-16_build-a-weather-station-using-google-cloud-iot-core-and-mongooseos/. [Último acceso: Mayo 2024].

[17] E. R. B. VEGA, *DISEÑO DE UNA ARQUITECTURA USANDO EL PROTOCOLO MESSAGE QUEUE TELEMETRY TRANSPORT (MQTT) SOBREPLATAFORMAS DE BAJO COSTE, PARA MONITORIZACIÓN DE PROCESOS INDUSTRIALES,* Riobamba, 2020.

[18] H. E. G. Rico y M. A. G. Ramón, *IMPLEMENTACIÓN DE PROTOCOLO DE COMUNICACIÓN PARA PRUEBAS FUNCIONALES ENTRE SISTEMA EMBEBIDO DE BAJO COSTO Y ELECTRODOMÉSTICOS,* Tabasco, 2020.

[19] C. Arana, *REDES NEURONALES RECURRENTES: ANÁLISIS DE LOS MODELOS ESPECIALIZADOS EN DATOS SECUENCIALES,* Buenos aires, 2021.

[20] D. Science, «DataScientest,» 20 Mayo 2022. [En línea]. Available: https://datascientest.com/es/memoria-a-largo-plazo-a-corto-plazo-lstm.

[21] S. JORDI, TECH pedia, Czech republic, 2016.

[22] Á. Perles, Sistemas Informáticos Industriales Comunicación serie RS 232, Valencia: Universidad Politecnica de valencia , 2018.

[23] D. X. V. Redrobán y J. E. D. Pacheco, «“ANALIZADOR DE PROTOCOLO SPI”,» Guayaquil, 2013.

[24] K. Hemmanur, «Inter-Integrated Circuit,» Michigan , 2009.

[25] H. Zhou, *A Wireless Sensor Network Air Pollution Monitoring System,* 2022.

[26] J. C. Mendoza, *Desarrollo de un sistema de monitoreo meteorológico basado en IoT para la predicción de eventos climáticos extremos,* 2023.

[27] L. Garcia, *Sistema de alerta temprana para eventos meteorológicos extremos utilizando tecnologías IoT y aprendizaje automático,* 2023.

[28] M. A. Rodriguez, *Implementación de una red de sensores IoT para la monitorización ambiental en zonas rurales,* 2022.

[29] Amazon, «Amazon,» 2018. [En línea]. Available: https://www.amazon.com/-/es/estaci%C3%B3n-meteorol%C3%B3gica-profesional-monitoreo-compatible/dp/B01N5TEHLI/ref=sr_1_1?crid=1DO3MCYC82AZ0&dib=eyJ2IjoiMSJ9.bdW6euTdpENcP7gi-DPs8GEHVRYWM7tHHqcu0FxWfFPstg7zboYeCVMZ4zQ9sOXkmpE-OVKwTBPmacksqAIO3wp4-HN-hJ0J7urL.

[30] T. D. Electronics, «Dynamo electronics,» 2021. [En línea]. Available: https://www.dynamoelectronics.com/tienda/anemometro-sensor-velocidad-del-viento-analogico/.

[31] Made-in-china, «Made-in-china,» 1998-2024. [En línea]. Available: https://china-sentec.en.made-in-china.com/product/OJjpdZfGfFVn/China-Sem228p-RS485-Photosynthetically-Active-Radiation-Sensor-PAR-Sensor-for-Plant-Growth-Meap-Rssurement.html.

[32] S. E. LTDA, «SIGMA Electronica,» [En línea]. Available: https://www.sigmaelectronica.net/producto/esp-32/.

[33] J. Novillo-Vicuña, D. H. Rojas, B. M. Olivo, J. M. Ríos y O. C. Villavicencio, ARDUINO Y EL INTERNET DE LAS COSAS, Alzamora, 2018.

[34] M. IFTTT, «SCRIPT,» 2019. [En línea]. Available: https://es.scribd.com/document/429407351/Manual-IFTTT-Es.

[35] J. L. Izquierdo, Codifica en Python volumen 1, quito : Universidad politecnica salesiana, 2020.

[36] P. M. C. RAMOS, «CARACTERIZACIÓN DE ENERGÍA ELÉCTRICA DE CLIENTES RESIDENCIALES POR MEDIO DE IOT,» Quito, 2020.

[37] D. A. M. L. B. V. C. E. A. A. L. F. a. S. R. Aponte-Roa, «Evaluation of a low-cost, solar-powered weather,» de *International Instrumentation and Measurement Technology Conference (I2MTC)*, 2018.

[38] h. L. X. Z. Y. L. Guohe Huang, *Impacts of climate change on photovoltaic energy potential a case study of china.,* 2020.

[39] C. N. A. L. F.D Vasquez, *Evaluación del efecto de las Variables Meteorológicas en el desempeño Térmico de una Edificación residencial a Partir de Datos Monitoreados,* Quito, 2020.

[40] R. L. L. Cristhian Garcia A, *Implementacion de prototipo IOT para la monitorizacion ambiental radiacion solar utilizando hardware de bajo costo,* 2024.

[41] S. A. y. G. S. Prisma Megantoro, *Estación meteorológica basada en IoT con medición de la calidad del aire usando ESP32 para estudio de condiciones aéreas ambientales,* Indonesia: Universidad Airlangga, 2021.

[42] INAMHI, *Instituto Nacional de Meteorología e Hidrología,* 2019.

[43] C. &. M. J. Vicente, *CONSTRUCCIÓN DE UNA ESTACIÓN COMPUTACIONAL PARA LA EVALUACIÓN DEL RECURSO EÓLICO Y SOLAR EN EL CAMPUS JOSÉ RUBÉN ORELLANA.,* 2019.

[44] G. Echeto, «PROTOCOLO IEC-104/VSAT APLICADO AL SEGUIMIENTO Y CONTROL DESUBESTACIONES ELECTRICAS,» *Revista Electronica de Estudios Telematicos,* vol. 8, 2009.