MODELO MATEMÁTICO PARA PREDICCIÓN DEL CLIMA USANDO COMPRESSIVE SENSING Y DEEP LEARNING

Andrés Alfonso Varelo Silgado

Universidad de Pamplona

Facultad de ingenierías y Arquitectura

Departamento de Ingeniería Eléctrica, Electrónica, Sistemas y Telecomunicaciones

Ingeniería Electrónica

Pamplona, Norte de Santander

2020

1

MODELO MATEMÁTICO PARA PREDICCIÓN DEL CLIMA USANDO COMPRESSIVE SENSING Y DEEP LEARNING

Andrés Alfonso Varelo Silgado

Tesis para pregrado

Tutor

Luis Enrique Mendoza

Universidad de Pamplona

Facultad de ingenierías y Arquitectura

Departamento de Ingeniería Eléctrica, Electrónica, Sistemas y Telecomunicaciones

Ingeniería Electrónica

Pamplona, Norte de Santander

2020

2

Nota de aceptación

_____
_____
_____
_____
_____
_____
_____

Jurado 1

Jurado 2

Jurado 3

Pamplona, Norte de Santander

17 de junio de 2020

AGRADECIMIENTOS

Quiero agradecer a todas las personas que ha creído en mi durante mi proceso de formación y de elaboración de este trabajo de grado, en especial al Ingeniero Luis Enrique Mendoza por confiar en mis capacidades apoyando esta idea de tesis, además de brindar parte su conocimiento en este camino.

Quiero expresar mi gratitud a toda la planta de docente del programa de Ingeniería Electrónica de la Universidad de Pamplona por brindarme su enseñanza para una adecuada formación como profesional.

Agradecimientos a todos mis conocidos y familiares por aceptarme, tener la paciencia para entender y ser apoyo en esta trayectoria.

A todos, muchas gracias.

DEDICATORIA

A Dios
Por quién siempre profesaré mi fe, quien ha sido mi auxilio en los momentos difíciles, dándome su inmerecida gracia e inexplicablemente el conocimiento que me permitido llegar hasta aquí.

A mi madre Cenia Lucía Silgado de la Rosa
Por ser esa persona determinada con la capacidad de tener grandes responsabilidades y formarme con los principios que hoy defiendo, por creer siempre en mi, por apoyarme en cada decisión que he tomado, eres gestora de todo este proceso

A mi Padre Alfredo Varelo Morales
Por ser la imagen de sabiduría durante toda mi vida, por enseñarme muchas cosas y unos principios morales inamovibles, por no dudar de mi en ningún momento.

A toda mi familia
Por darme amor, respeto, cariño y corrección durante todo este tiempo, por siempre creer en mi, son lo más importante que tengo.

A mis amigos en Pamplona
Por aceptarme y ser mi familia en los momentos lejos de casa, brindarme su conocimiento y apoyo en cada instante de este camino de formación.

CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

RESUMEN

La predicción del estado del tiempo es un aspecto importante en todos los sectores industriales desde la extracción de materias primas, su procesamiento hasta la obtención del producto final. Por este motivo, es necesario tener control del estado del tiempo conociendo la tendencia. Este documento describe el desarrollo de un modelo matemático usando Compressive Sensing para la predicción del clima tomando el promedio diario de variables como Temperatura, Humedad Relativa, Velocidad del viento, Precipitación e insolación desde 2015 hasta 2020 de la base de datos POWER NASA , la técnica implementada fue capaz de reducir la longitud de series de tiempo de 130 puntos hasta 26, 39 y 52 comprimiendo a un 80%, 70%, 60 % respectivamente mediante el uso de las matrices identidad unitaria y la transformada inversa discreta del coseno de un arreglo diagonal para la compresión de los datos, el aprendizaje fue implementando haciendo uso de las librerías Keras y Tensorflow del entorno de desarrollo Python mediante una red neuronal multicapa con optimizador Adam para una mayor velocidad en la extracción de patrones. A nivel general se obtuvieron porcentajes de precisión mejores para la temperatura, humedad relativa, velocidad del tiempo e insolación, pero la precipitación presentó las estadísticas más bajas, se afirma que la técnica cumple las expectativas porque el tener solo 4 años de histórico es muy poco para los métodos tradicionales, pero Compressive Sensing los potencializa para una adecuada predicción de los parámetros.

ABSTRAC

The weather forecast is an important item in all industry, from extract the raw matery, its processing to get the final product. Thus, it's necessary control the weather knowing its trend. This document describe development a mathematic model using Compressive Sensing to predict the weather with the daily mean of variables like Temperature, Relative Humidity Wind Speed, Precipitation and Insolation from 2015 to 2020 employing POWER NASA data base, this technique was able to reduce time series length with 130 points until 26, 39 and 52 compressing 80%, 70% and 60% respectively utilizing a Identity matrix and its Discrete Cosine Transform to compact the information, the learning was development with Keras and Tensorflow in Python environment through a Multilayer Neural Network with Adam optimizer to faster extract features. Generally, the best accuracy was to Temperature, Relative Humidity, Wind Speed and Insolation but the precipitation got the worst statistics. It affirmed this model comply with expectative because 4 historic years is very low for the traditional methods, but Compressive Sensing strengthened to good forecast.

INTRODUCCIÓN

El estado del tiempo es uno de los aspectos determinantes en el desarrollo de la sociedad ya que la comunión y entendimiento de las personas con su entorno posibilita una planeación adecuada de sus actividades esto se ve reflejado en el mundo agrícola, espacio regido directamente por las condiciones atmosféricas para una producción adecuada de los alimentos que abastecen a la comunidad urbana esto ha llevado al hombre a querer tener el control de todos los factores que puedan contribuir al daño de los cultivos de esta necesidad conectadas con situaciones urbanas e incluso deportivas correspondientes a la planeación de actividades y eventos, se dio origen al estudio de la predicción del estado del tiempo implementado mediante herramientas matemáticas conocidas basadas en datos históricos dado que el tiempo se divide en temporadas con condiciones características a lo largo de un año que posteriormente se denominaron estaciones climática, a pesar de que las técnicas en su mayoría regresivas son confiables pues presentan inconvenientes a la hora de enfrentarse a parámetros tan complejos como los presentados por el clima, ya que la reciente intervención de la mano destructiva de la humanidad ha generado cierta aleatoriedad en el orden y duración de los periodos de lluvias, sequias, heladas, etc. Además de hacer aun más críticas dichos cambios drásticos tienen grandes repercusiones en el medio ambiente, las dificultades anterior hacen más compleja la predicción del estado del tiempo para un hombre e incluso para las técnicas conocidas lo que hace necesario implementar métodos avanzados que brinden robustez y confiabilidad donde incursiona el uso de inteligencia artificial especializada que se encargue de percibir patrones característicos de forma automática y establecer una estimación adecuada, por tanto este documento explica el desarrollo de un modelo matemático para predecir el estado del tiempo buscando contribuir a la bibliografía existente y abrir un camino de nuevos trabajos similares.

PLANTEAMIENTO DEL PROBLEMA

El cambio climático es uno de los problemas que más efectos produce en la sociedad actual desencadenando situaciones anormales tales como el fenómeno del niño el cual ha afectado la precisión en el dictamen del estado del tiempo. Dicho proceso, en su curso natural no presenta un comportamiento sencillo y se ha incrementado su aleatoriedad con distintas situaciones de temperaturas extremas tales como sequías, heladas e inundaciones en tiempos no previstos, además actualmente en nuestro país se determina un pronóstico muy globalizado, descuidando variaciones en regiones específicas, ocasionando incertidumbre en la población y masivos daños en el sector agrícola, entorpeciendo la producción de la zona en específico.

De lo anterior, se puede concluir que el Clima no presenta patrones estáticos y marcados y que el cambio climático aumenta su aleatoriedad cada vez más, haciendo más inciertas las condiciones extremas. Entonces se plantea el siguiente interrogante, ¿Es posible mediante un modelo matemático de extracción de patrones e inteligente predecir con más exactitud el estado del tiempo en una zona determinada?

JUSTIFICACIÓN

La predicción del estado del tiempo es una información primordial para el sostenimiento de la sociedad ya que es determinante en la producción de alimentos y por tanto en el desarrollo económico de la región. Por otro lado, es importante optimizar la precisión del diagnóstico de las condiciones meteorológicas proceso realizado normalmente con técnicas estadística acompañadas de complejas ecuaciones y consideraciones con ciertos grados de error, la incursión de nuevos enfoques como la inteligencia artificial, podría automatizar la metodología tradicional creando un sistema que esté constantemente aprendiendo y reconociendo los patrones aleatorios y extraños del estado del tiempo, para así brindar un soporte confiable a los campesinos del país y así se puedan realizar las estrategias de contingencia adecuadas en las fechas acertadas reduciendo las perdidas, disminuyendo los recursos implementados en las actividades de prevención.

OBJETIVOS

OBJETIVO GENERAL

Desarrollar un modelo matemático para la predicción del clima usando Compressive Sensing y Deep Learning.

OBJETIVOS ESPECÍFICOS

1. Adquirir las variables ambientales necesarias y adecuadas de manera automática que se relacionen con la variabilidad.

2. Definir la metodología de extracción de patrones y del proceso de entrenamiento del sistema inteligente con el fin de conseguir las mejores características en el modelo de predicción de variables ambientales basado en Compressive Sensing y Deep Learning.

3. Desarrollar una plataforma capaz de modelar la información entrarte con el fin de mostrar resultados de futuras características del estado del tiempo.

4. Realizar pruebas con información aleatoria no conocida para verificar la robustez del sistema

CAPÍTULO I

DESARROLLO TEÓRICO

En este capítulo se describen los conceptos y proyectos de referencia más importantes tomados y utilizados en el desarrollo de este trabajo.

1.1 ESTADO DEL ARTE

El estudio del clima utilizando técnicas inteligentes es un trabajo que se ha venido realizando en los últimos años, para obtener una mayor precisión a la hora de dar un dictamen con respecto a las condiciones meteorológicas de un lugar determinado. Haciendo una revisión bibliográfica es posible afirmar que el contexto internacional ha brindado una cantidad importante de aportes. A continuación, se presentan algunos resultados relacionados con el trabajo propuesto.

El trabajo **Modelado y Predicción del Fenómeno El Niño en Piura, Perú mediante Redes Neuronales Artificiales usando Matlab** elaborado por Jiménez et al. (2018) en la Universidad Nacional de Piura, aplicando redes neuronales a datos como precipitación, temperaturas superficiales de mar y velocidad de los vientos alisios para modelar y predecir el fenómeno del niño con 6 meses de antelación obteniendo un buen desempeño con un acierto de 98.4 % en el entrenamiento y 100% en la predicción del primer semestre de 2016 [1].

También la Universidad de Colima y la Universidad autónoma de Nayarit realizaron el **Sistema Multiagente para la predicción del clima para usos agrícolas** liderado por Fuentes et al. (2017) quienes elaboraron un sistema informático basado en inteligencia artificial para ofrecer pronósticos obteniendo interfaz gráfico amigable el usuario capaz de establecer un dictamen fácil de interpretar con la información meteorológica en el día indicado [2].

En Bolivia más exactamente en la Universidad Católica Boliviana San Pablo en 2018, Mercado F. en su tesis de Maestría llamada **Máquina de aprendizaje extremo para la predicción temprana de heladas en valles meso-térmicos de Bolivia**. Propone la creación de una máquina de aprendizaje extremo mediante el algoritmo de Huang sobre una red neuronal de una capa con propagación hacia adelante, como base para predicción temprana de heladas meteorológicas. El estudio se realizó el en el valle de alto y central de Cochabamba con datos de 8 estaciones meteorológicas de la institución meteorológica de la nacional SENAMHI. Formando dos dataset una para

entrenamiento y otro para verificación. El primer conjunto de datos se tomó de 5 estaciones desde diciembre de 2012 hasta agosto de 2016 aproximadamente para 1360 días con un total de 178450 mediciones realizadas cada 15 minutos. El conjunto de datos de validación se accedió a los datos de 2 estaciones entre octubre de 2016 y noviembre de 2017 teniendo 378 días con 46288 medidas, obteniendo que el algoritmo es más rápido un 203% en el entrenamiento y 220% en validación con factores de correlación de Pearson de 0.9741 en entrenamiento y 0.9894 en validación por lo que el desarrollo es una excelente predictor de heladas con altos niveles de confianza y adecuados tiempos de respuesta [3].

Los desarrollos de Kaur y Maqsood en su trabajo **An ensemble of neural networks for weather forecasting, Neural Comput & Applic** (2004), describen modelos de predicción de 24 horas de temperatura, velocidad del viento y humedad relativa. El entrenamiento y evaluación se realizan por separado para cada estación. Los autores compararon el Perceptrón Multicapa (MLP), Red de Función de Base Radial (RBFN), modelo de Hopfield (HFM) y Red Neuronal Recurrente de Elmann (ERNN). MLP fue entrenado con Retropropagación y función de activación sigmoide logística en la capa oculta con 72 neuronas mientras que RBFN usó aprendizaje natural no supervisado y función activación Gaussiana (en dos capas ocultas de 180 neuronas), ambas usaron una salida lineal. La precisión se midió con porcentaje promedio de error absoluto (MAPE). RBFN tuvo un mejor rendimiento no en términos de precisión sino en costo computacional. Para invierno y primavera la humedad tuvo el menor MAPE, para verano y otoño la temperatura tuvo una mejor predicción [4].

El trabajo. **A feature based neural network model for weather forecasting, World Academy of Science, Engineering and Technology** (2007) descrito por Sanjau Marthur se enfoca en predicción de la temperatura máxima, mínima y humedad relativa analizándolas como series de tiempo. Usando un Red multicapa Retroalimentada con aprendizaje por Retropropagación. Los parámetros de entrada fueron estáticos para predicción adecuada de temperaturas máximas y mínimas se tomó un periodo de 15 semanas. Las características usadas fueron promedio móvil, promedio móvil exponencial, oscilación, razón de cambio y tercer momento. El error fue menor al 3%. El principal resultado es que los parámetros principales pueden ser usado para extraer patrones. Las mejores características fueron promedio móvil, promedio móvil exponencial, oscilación, razón de cambio y momentos. Asimetría y Kurtosis no funcionaron bien [5].

En **Application of artificial neural networks for temperatura forecasting** (2007) se desarrolló un sistema para predecir a corto tiempo por Hayati, siendo a su vez MLP con 6 capas ocultas con función de activación sigmoide y linear

para la salida. Se usó algoritmo de gradiente conjugado escalizado para entrenamiento. Como parámetros de entrada se tomaron mediciones cada 3 horas de velocidad del viento, dirección del viento, bulbo seco, temperatura, temperatura de bulbo seco, humedad relativa y punto de rocío, presión, visibilidad, cantidad de nubes, ráfaga de viento, temperatura promedio, temperatura máxima y mínima, precipitación, humedad promedio, presión promedio, evaporación [6]

Una red MLP con 3 capas es presentada por Baboo en **An eficient weather forecasting system using Artificial neural network**, (2010) donde el error fue muy poco. Las entradas difieren entre los que se seleccionaron presión atmosférica, temperatura atmosférica, humedad relativa, velocidad del viento y dirección. Se entrenó mediante Retropropagación. Las predicciones son restringidas a un área determinada [7].

Aunque la mayoría de los enfoques usaron MLP Caltagirone, S. en su tesis de maestría **Air temperature perdición using evolutionary artificial neural networks** (2001), usa una red neuronal evolucionaria en combinación con algoritmos genéticos para predecir la temperatura por día. Los datos de entrada fueron mensuales y diarios donde destacan precipitación, temperatura máxima y mínima, temperatura máxima y mínima del suelo, humedad relativa máxima y mínima, radiación solar y velocidad del viento. Con una precisión del 79.49% para grados de error. El entrenamiento fue realizado con algoritmo de Retropropagación. Al autor asume que se puede mejorar el rendimiento con una mayor cantidad de datos [8].

1.2 CLIMA

Se conoce al clima como el grupo de condiciones atmosféricas aleatorias determinados por los estados y evoluciones del tiempo, durante intervalos temporales en una región determinada, controlados por los factores forzantes, factores determinantes y la interacción de los elementos del sistema climático, los cuales son: atmosfera, hidrosfera, criósfera, biosfera y antropósfera. Debido a que el clima está relacionado con las condiciones de la atmosfera, el mismo se describe a partir de variables como la temperatura y precipitación que son de tipo atmosféricos [9]

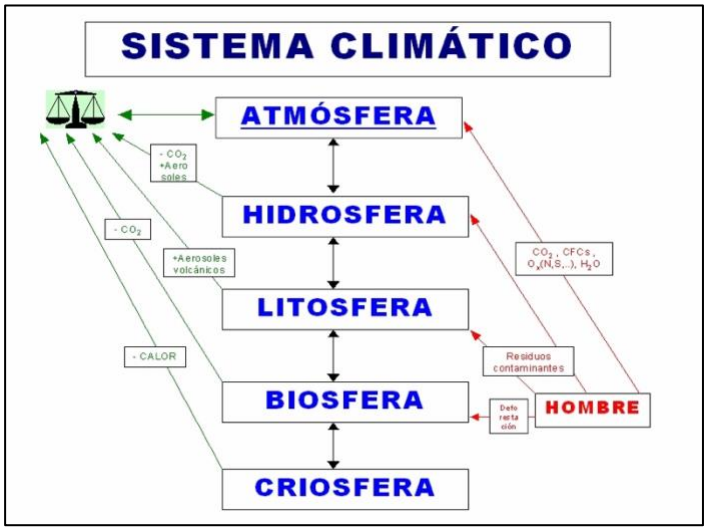


Figura 1.1 Sistema Climático *[10]*

1.2.1 Variabilidad climática y predicción

En la actualidad la emulación de la evolución de la atmosfera se hace mediante modelos numéricos de circulación general atmosféricas que resuelven ecuaciones de rigen las condiciones de frontera, enfocadas a la respuesta del clima a forzamientos como variaciones de la radiación solar, temperatura superficial del mar, concentraciones de GEI, entre otros. Y entonces obtener los estados futuros de la atmósfera. Igualmente, mediante técnicas estadísticas es posible inferir los estados atmosféricos próximos usando series de registros históricos en un sitio particular [11].

Los modelos numéricos hacen representación tridimensional de la circulación general de la atmosfera del planeta, conocidos como Modelos climáticos tienen una resolución horizontal entre 250 y 600 km, mientras que verticalmente, entre 10 y 20 capas de la atmosfera. El inconveniente de estos es la poca precisión en áreas muy extensas. Por lo que se opta por modelos climáticos regionales para mejorar la resolución en una región limitada.

1.3 Inteligencia Artificial

Existe algunos campos de la tecnología rodeados de un halo de misterio tal como la Inteligencia Artificial o Machine Learning, asumiendo que para abordar un proyecto se necesita una montaña de conocimiento, pero es una rama de la ciencia que con el debido trabajo se puede comprender. (Libro Machine Learning).

La inteligencia Artificial es la rama de las Ciencias de la computación que estudia software y hardware para simular el comportamiento y compresión humanos. La meta final de esta simular la inteligencia humana en una máquina creando modelos capaces de tener consciencia y sentimientos aproximados a los del hombre [12].

Existen varias técnicas de inteligencia artificial donde se encuentran:

- Machine Learning o aprendizaje automático
- Lógica Difusa o Fuzzy logic
- Vida artificial
- Sistemas expertos
- Data Mining o Minería de datos
- Redes Bayesianas
- Redes Neuronales Artificiales
- Sistemas reactivos
- Sistemas basados en reglas
- Razonamiento basado en casos
- Técnicas de Representación de Conocimiento
- Redes Semánticas
- Lingüística computacional
- Procesamiento de lenguaje natural

1.3.1 Machine Learning

El Machine Learning (ML) es una técnica de inteligencia artificial para usar computadoras para predecir cosas basado en observaciones pasadas, se suelen tomar datos de un proceso determinado para entonces crear un programa capaz de analizar dichos datos y predecir condiciones futuras.

La lógica de programación en Machine Learning es distinta a la usualmente usada. Una sección de código tradicional está diseñada para tomar unas entradas, aplicar varias reglas y generar una salida. El algoritmo opera de acuerdo a la implementación del programador en sus líneas de código, donde le programador debería entender los datos para indicar el problema y así desarrollar un código que las entienda, en cambio en ML el programador suministra datos un tipo especial de datos al algoritmo y que el algoritmo descubra la reglas, es decir, un programa con la facultad de entender toda la complejidad de los datos por ellas mismas, construyendo un modelo del sistema basado en la información, es posible ***entrenar*** el modelo para mejorar su estructura, y así luego llamar un modelo para hacer predicciones [13].

1.3.2 Deep Learning

El machine Learning tiene muchos enfoques donde uno de los más populares es el *Deep Learning* – Aprendizaje Profundo. El cual está basado en una idea simplificada de cómo funciona el cerebro humano. En esta técnica una red de neuronas simuladas las cuales están representadas por arreglos de números, es entrenada para obtener un modelo de relaciones entre varias entradas y salidas. Existen distintas arquitecturas o arreglos neuronales útiles para diferentes tareas. Por ejemplo, algunas arquitecturas sobresalen para obtener patrones de imágenes, otras para predecir el siguiente valor de una secuencia. La **Figura *1.2*** indica la estructura de una neuronal donde las neuronas verdes indican las *p* entradas, las moradas son las capas ocultas que se encargan del extraer características y las rojas son las 9 salidas [13].

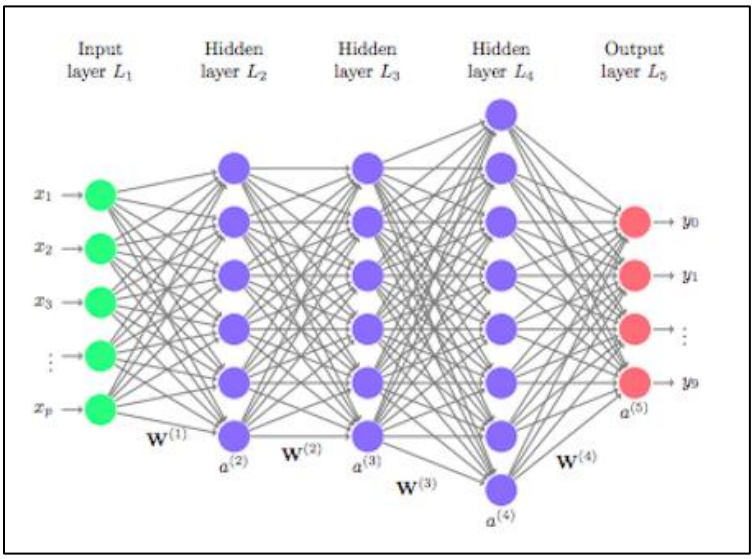


Figura 1.2 Estructura de Red Neuronal (Deep Learning) *[14]*

1.3.3 Neurona Biológica

Son pequeñas neuronas que se ubican en el sistema nervioso de hombre encargadas de activar o inhibir la actividad eléctrica de este, su principal función consiste en recibir estímulos eléctricos y conducirlos hacia otras neuronas estímulo conocido como potencial de acción, mediante este proceso es posible mover extremidades, sentir dolor e incluso soñar [15].

La **Figura *1.3*** describe las partes de una neurona biológica las cuales se profundizan en la sección 1.2.4.

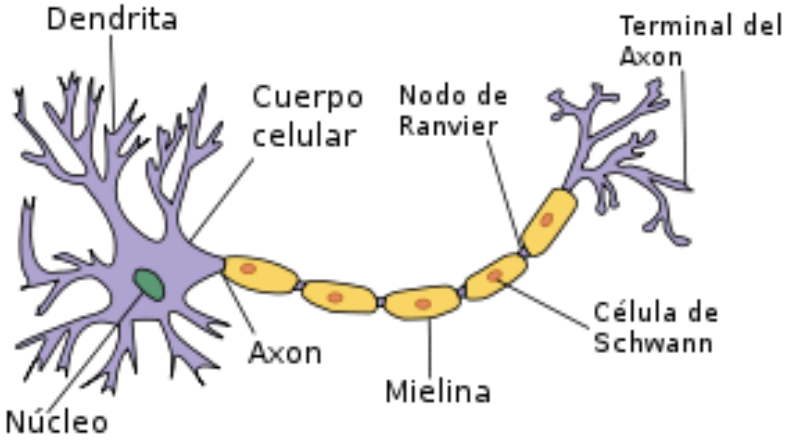


Figura 1.3 Neurona Biológica *[16]*

1.3.4 Partes neurona biológica

1.3.4.1 Núcleo

Es el centro de la neurona donde se lleva a cabo toda la actividad metabólica, se generan nuevas moléculas y se mantiene viva la célula, se transmite información entre neuronas se debe producir grandes cantidades de proteínas. En este lugar reposan las proteínas fibrales, que forman el citoesqueleto que provee el mecanismo para el transporte de moléculas [17].

1.3.4.2 Dendritas

Son prolongaciones en forma de ramas de tamaño reducido que nacen del cuerpo de neuronal y reciben estímulos y alimenta a la célula, incorporando a las espinas dendríticas donde se realiza la sinapsis proceso que hace posible la transmisión de potenciales de acción [17].

1.3.4.3 Axón

Es una prolongación de la célula encargada de transportar el potencial de acción del cuerpo de la neurona hacia otra neurona, pueden estar cubiertos de mielina la cual potencia la rapidez con la que se transmiten de potenciales de acción [17].

1.3.4.4 Potencial de Acción

Es la señal eléctrica que viaja a través del axón o núcleo con las características que siempre posee el mismo potencial a lo largo de toda la longitud del mismo

lo que se explica por el hecho que a traviesa unos canales iónicos que impiden disminuir este valor. La forma de onda del mismo se ve en la **Figura *1.4*** donde se puede ver que la neurona está a una tensión de -70 mv en estado de reposo pero cuando se recibe un estímulo mayor a -55 se activa este potencial de acción durante el periodo refractario la célula se bloquea y no es posible volver a recibir un estímulo [18].

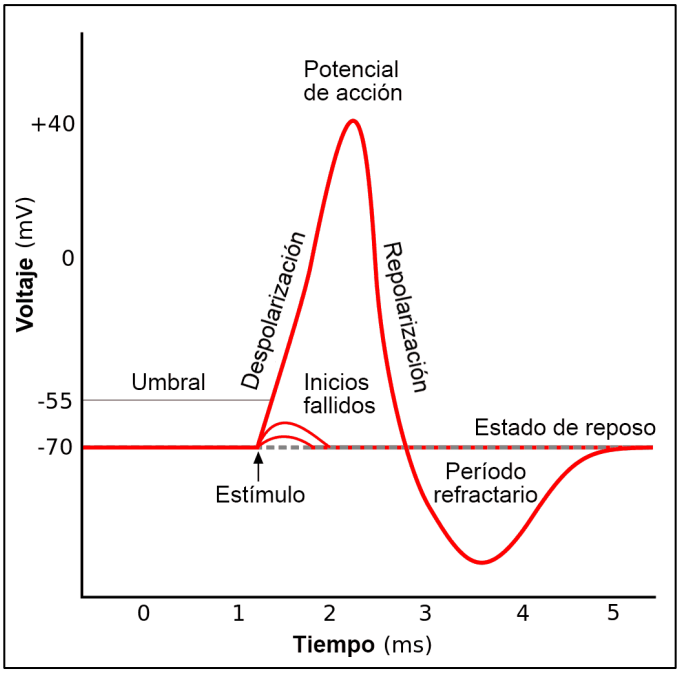


Figura 1.4 Potencial de Acción *[19]*

1.3.5 Funcionamiento de Neurona Biológica

El funcionamiento de una neurona está basado en la sinapsis que a grandes rasgos concibe el hecho de cambio de información de entre células, el proceso se compone de cuatro etapas [18];

Etapa 1: El potencial de acción correspondiente llega a la neurona pre sináptica estimulando la liberación de vesículas con neurotransmisores hacia la célula post-sináptica.

Etapa 2: Se activan los canales iónicos para que entren los iones ya sean positivos o negativos, los primeros crean potenciales sinápticos excitatorios y los segundos potenciales sinápticos inhibitorios.

Etapa 3: La intensidad de cada potencial dependerá del número de transmisores que se liberen, esto es llamado la intensidad de la conexión lo que también se conoce como Peso Sináptico

Etapa 4: Las otras neuronas estimulan las dendritas mediante la sinapsis creando potenciales tantos excitatorios como inhibitorios sumando cada uno de estos y enviándolo a las siguientes neuronas repitiendo el ciclo (Etapa 1)

1.3.6 Neurona Artificial

Para recrear una neurona mediante algoritmos se debe recordar que ésta a grandes rasgos recibe los potenciales en las dendritas mediante la sinapsis creando potenciales tantos excitatorio como inhibitorio dependiendo de la intensidad de la conexión lo que se puede asociar a un ***producto*** estos a su vez se ***suman*** en el cuerpo de la neurona creando un potencial propio que si supera un umbral determinado será excitatorio o inhibitorio [18].

Se consideran R entradas o patrones $[P_1, P_2, \dots, P_R]$ a los cuales se les asocia un peso determinado $[W_{11}, W_{12}, \dots, W_{1R}]$ que corresponde a la intensidad de la conexión entre la entrada y su receptor, además de una polarización unitaria *b1* emulando los -70 mV de la neurona biológica, esta sumatoria da la entrada neta como se en la ecuación (1.1).

$$n_1 = P_1 W_{11} + P_2 W_{12} + \cdots + P_R W_{1R} + b_1 \qquad (1.1)$$

A dicha sumatoria se le aplica una función de activación lo que determina que la neurona se active o no, generado una salida a_1.

Donde es posible definir los pesos sinápticos y los patrones como unas vectores columnas de la forma.

$$w = \begin{bmatrix} W_{11} \\ W_{12} \\ \vdots \\ W_{1R} \end{bmatrix} \quad p = \begin{bmatrix} P_1 \\ P_2 \\ \vdots \\ P_R \end{bmatrix}$$

Donde se puede sintetizar la activación de la neurona como se ve en la ecuación (1.2):

$$a_1 = f(\boldsymbol{w^T p} + b1) \qquad (1.2)$$

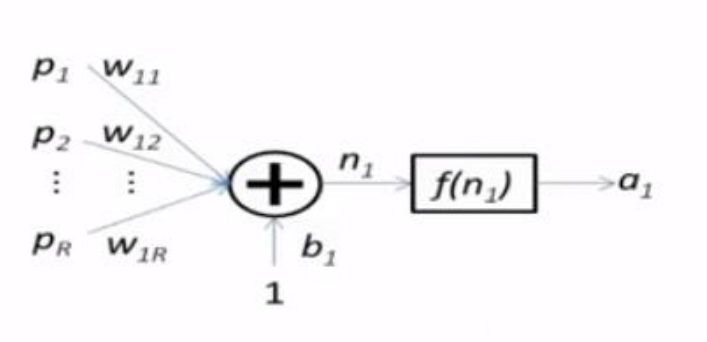


Figura 1.5 Estructura de Neurona Artificial *[18]* **.**

1.3.7 Red Neuronal Monocapa

Se consideran S neuronas y R entradas o patrones, donde a cada neurona se le asocian todas las entradas con unos pesos respectivos y cada neurona tendrá una salida de acuerdo a su característica específica como se observa en la **Figura *1.6* Estructura de Red Monocapa** de igual forma es posible compactar el sistema mediante ecuaciones matriciales tal como lo ilustra la ecuación (1.3) [18]

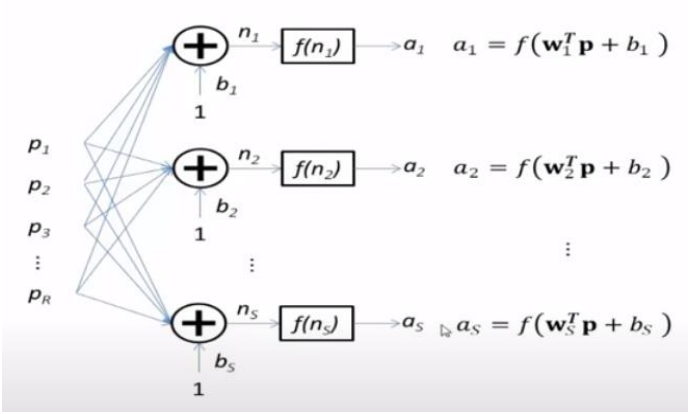


Figura 1.6 Estructura de Red Monocapa

$$a = f(\boldsymbol{wp} + \boldsymbol{b}) \qquad (1.3)$$

Donde,

$$p = \begin{bmatrix} P_1 \\ P_2 \\ \vdots \\ P_R \end{bmatrix} \quad b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_R \end{bmatrix} \quad n = \begin{bmatrix} n_1 \\ n_2 \\ \vdots \\ n_R \end{bmatrix} \quad p = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_R \end{bmatrix} \quad w = \begin{bmatrix} W_1^T \\ W_2^T \\ \vdots \\ W_R^T \end{bmatrix} = \begin{bmatrix} W_{11} & W_{12} & \dots & W_{1R} \\ W_{21} & W_{22} & \dots & W_{2R} \\ \vdots & \vdots & \ddots & \vdots \\ W_{S1} & W_{S2} & \dots & W_{SR} \end{bmatrix}$$

Donde los pesos están dados por la expresión w_{ij} donde i corresponde a la neurona y j a la entrada.

La mejor forma de comprender las expresiones matriciales es conociendo lo que pasa con las dimensiones de cada vector, tal como en la **Figura *1.7*** donde se ve una serie de patrones ***R*** expuestos a ***S*** neuronas generan una salida correspondiente al número de neuronas

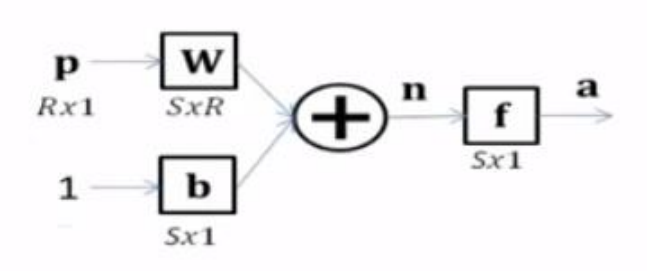


Figura 1.7 Dimensiones vectoriales en una red monocapa *[18]*

1.3.8 Red Neuronal Multicapa

Existen problemas que no son linealmente separables dentro del estudio de redes neuronales par eso se suele juntar en serie algunas capas individuales de neuronas que ofrezca una mejor solución a dicho problema, lo anterior se conoce como redes neuronales multicapa y se ilustra en la **Figura *1.8*** [18],

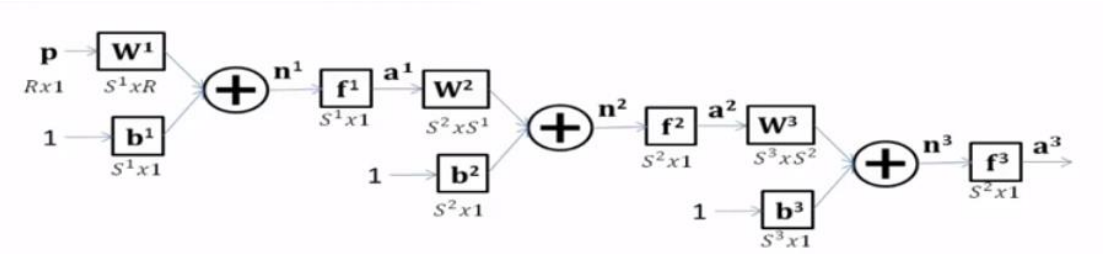


Figura 1.8 Red Neuronal Multicapa *[18]*

Se puede observar que el número de patrones ***R*** se modifica en función de las neuronas de la capa 1, S_1, luego el número de neuronas de esta capa se convierten en las entradas de la capa 2, que a su vez tendrá una salida que alimentará a la tercera capa, la salida total de la red neuronal está determinada por la última capa, en función de las anteriores.

$$a_3 = f^3(w^3 f^2(w^2 f^1(w^1 p + b^1) + b^2) + b^3) \quad (1.4)$$

w^i = Pesos de la capa i
b^i = Polarizaciones de la capa i
f^i = Función de activación de capa i
a^i = Salida de capa i

1.3.9 Aprendizaje general de Red Neuronal

Se hace necesario un algoritmo capaz de ajustar automáticamente los pesos y la polarización una capa de neuronas, como se muestra en la **Figura *1.9*** donde ***p*** son los patrones o entradas y ***t*** son los targets o salidas correspondiente [18].

Iniciar aleatoriamente **w,b**
Desde épocas=1 a N_{epocas}
    Desde q=1 a $Q_{repetir}$
        $a_q = f(wp_q + b)$
        $e_q = t_q - a_q$
        $w = w + e_q t_q^T$
        $b = b + e_q$
    Fin
Fin

Figura 1.9 Algoritmo de Aprendizaje General

1.3.10 Funciones de Activación

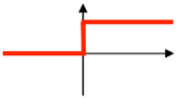
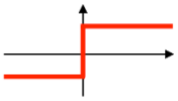
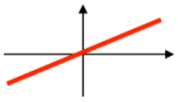
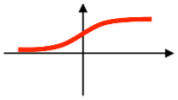
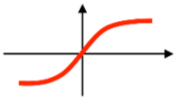
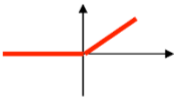
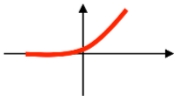
Activation function	Equation	Example	1D Graph
Unit step (Heaviside)	$\phi(z) = \begin{cases} 0, & z < 0, \\ 0.5, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Sign (Signum)	$\phi(z) = \begin{cases} -1, & z < 0, \\ 0, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Linear	$\phi(z) = z$	Adaline, linear regression	
Piece-wise linear	$\phi(z) = \begin{cases} 1, & z \geq \frac{1}{2}, \\ z + \frac{1}{2}, & -\frac{1}{2} < z < \frac{1}{2}, \\ 0, & z \leq -\frac{1}{2}, \end{cases}$	Support vector machine	
Logistic (sigmoid)	$\phi(z) = \frac{1}{1 + e^{-z}}$	Logistic regression, Multi-layer NN	
Hyperbolic tangent	$\phi(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$	Multi-layer Neural Networks	
Rectifier, ReLU (Rectified Linear Unit)	$\phi(z) = max(0, z)$	Multi-layer Neural Networks	
Rectifier, softplus	$\phi(z) = \ln(1 + e^{z})$	Multi-layer Neural Networks	

Figura 1.10 Función de Activación *[20]*

1.3.11 Funciones de Pérdidas

Las pérdidas en el contexto de una red neuronal indica el error de una predicción determinando que tan lejos se está del modelo deseado y el método para medirlo es llamada función de pérdida. Las pérdidas son usadas para calcular los gradientes y a su vez este último son de ayuda para actualizar los pesos de la red neuronal y la forma como es entrenada [21].

1.3.11.1 Error Cuadrático Medio

La pérdida MSE es usada para la regresión. Como lo dice su nombre es calculado como el promedio de las diferencias entre el valor deseado y el predicho.

1.3.12 Gradiente Descendente

Es un método que es la base del aprendizaje automático tanto Machine Learning como en Deep Learning, el cual busca determinar los mejores coeficientes para disminuir la función de coste o error del modelo mediante el siguiente algoritmo [18]

```
Inicializar aleatoriamente θ
    Desde i=1 a N repetir
        θ= θ – α∇J(θ)
Fin
```

Figura 1.11 Algoritmo de Gradiente descendente

Donde,

α es la velocidad de aprendizaje

$\frac{dF}{dx}$ gradiente de la función de pérdida.

1.3.1 Algoritmo de Retropropagación

Es uno de los algoritmos más importantes de entrenamiento de redes neuronales enfocados al aprendizaje adecuado de modelos con múltiples capas, se meta esencial en encontrar la derivadas parciales o gradiente de cada capa que no es más que la derivada parcial de una función de coste o error con respecto a los pesos y las polarizaciones, parámetro necesario para actualizarlos haciendo uso del algoritmo de gradiente descendiente [22].

Particularmente esta técnica inicia con el error de la última capa luego retro-propaga el mismo hacia las capas anteriores hasta la inicial entrenando toda la red neuronal de manera adecuada.

Para un sistema de L capas considerando la ecuación (1.5) y (1.6)

$$z^L = w^L x + b^L \quad (1.5)$$

$$a(z^L) = f\_activacion(z^L) \quad (1.6)$$

$$C(a^L) = f\_coste(a^L) \tag{1.7}$$

Por tanto, se sabe que para encontrar la derivada de la función de coste se debe tener en cuenta la composición de funciones de la ecuación (1.7) que implica la misma al relacionarla con los pesos y las polarizaciones, obteniendo mediante la regla de la cadena las ecuaciones (1.8) y (1.9) para describir el gradiente de los pesos y las polarizaciones respectivamente.

$$\frac{\partial C}{\partial w^L} = \frac{\partial C}{\partial a^L} \cdot \frac{\partial a^L}{\partial z^L} \cdot \frac{\partial z^L}{\partial w^L} \tag{1.8}$$

$$\frac{\partial C}{\partial b^L} = \frac{\partial C}{\partial a^L} \cdot \frac{\partial a^L}{\partial z^L} \cdot \frac{\partial z^L}{\partial b^L} \tag{1.9}$$

Desarrollando cada derivada parcial, considerando una función de coste raíz cuadrática media definido por la ecuación (1.10) y su derivada parcial con respecto a la salida como la ecuación (1.11)

$$C(a^L) = \frac{1}{2}\sum (y_i - a_j^L)^2 \tag{1.10}$$

$$\frac{\partial C}{\partial a^L} = \left(a_j^L - y_i\right) \tag{1.11}$$

Tomando como ejemplo una función de activación sigmoide logística como se ve en las ecuaciones (1.12) y (1.13)

$$a^L(z^L) = \frac{1}{1 + e^{-z^L}} \tag{1.12}$$

$$\frac{\partial a^L}{\partial z^L} = a^L(z^L)\left(1 - a^L(z^L)\right) \tag{1.13}$$

Considerando la ecuación (1.5) como la suma ponderada z sus derivadas parciales con respecto a los pesos y polarizaciones se ven en las ecuaciones (1.14) y (1.15)

$$\frac{\partial z^L}{\partial w^L} = a_j^{L-1} \tag{1.14}$$

$$\frac{\partial z^L}{\partial b^L} = 1 \tag{1.15}$$

Haciendo el producto de las ecuaciones (1.11)(1.13)(1.14) y (1.11)(1.13)(1.15) se obtienen las ecuaciones (1.16) y (1.17) respectivamente

$$\frac{\partial C}{\partial w^L} = \left(a_j^L - y_i\right) a^L(z^L)\left(1 - a^L(z^L)\right).\, a_j^{L-1} \tag{1.16}$$

$$\frac{\partial C}{\partial b^L} = \left(a_j^L - y_i\right) a^L(z^L)\left(1 - a^L(z^L)\right).\, 1 \tag{1.17}$$

El término $\frac{\partial C}{\partial a^L}.\frac{\partial a^L}{\partial z^L}$ determina cuanto varía el error en función de la suma ponderada, indicando el grado de responsabilidad de las neuronas en la salida por tanto un valor elevado del mismo asume un gran compromiso de dicha capa son el error de la salida.

Esta derivada parcial es llamada ***error de neurona*** $\boldsymbol{\delta^L}$

Redefiniendo las ecuaciones (1.16) y (1.17)

$$\frac{\partial C}{\partial w^L} = \boldsymbol{\delta^L} a_j^{L-1} \tag{1.18}$$

$$\frac{\partial C}{\partial b^L} = \boldsymbol{\delta^L} \tag{1.19}$$

Para la capa L-1 se define las funciones de coste de los pesos y las polarizaciones como se observa en las ecuaciones (1.20) y (1.21)

$$\frac{\partial C}{\partial w^L} = \frac{\partial C}{\partial a^L}.\frac{\partial a^L}{\partial z^L}.\frac{\partial z^L}{\partial a^{L-1}}\frac{\partial a^{L-1}}{\partial z^{L-1}}\frac{\partial z^{L-1}}{\partial w^{L-1}} \tag{1.20}$$

$$\frac{\partial C}{\partial b^L} = \frac{\partial C}{\partial a^L}.\frac{\partial a^L}{\partial z^L}.\frac{\partial z^L}{\partial a^{L-1}}\frac{\partial a^{L-1}}{\partial z^{L-1}}\frac{\partial z^{L-1}}{\partial b^{L-1}} \tag{1.21}$$

Donde se sabe que

$$\frac{\partial C}{\partial a^L} \cdot \frac{\partial a^L}{\partial z^L} = \boldsymbol{\delta^L}\ por\ la\ ecuación\ (1.18) \tag{1.22}$$

$$\frac{\partial z^L}{\partial a^{L-1}} = \boldsymbol{w^L} \tag{1.23}$$

$$\frac{\partial a^{L-1}}{\partial z^{L-1}} = Derivada\ de\ función\ de\ activación \tag{1.24}$$

$$\frac{\partial z^{L-1}}{\partial w^{L-1}} = a_j^{L-2} \tag{1.25}$$

$$\frac{\partial z^{L-1}}{\partial w^{L-1}} = 1 \tag{1.26}$$

Donde las ecuaciones (1.22)(1.23)(1.24) definen el error de neurona de la capa L-1:

$$\frac{\partial C}{\partial a^{L-1}} = \boldsymbol{\delta^{L-1}} \tag{1.27}$$

Es posible resumir este algoritmo en 3 pasos

Calcular error de neurona de última capa ec. (1.22)

$$\frac{\partial C}{\partial a^L} \cdot \frac{\partial a^L}{\partial z^L} = \boldsymbol{\delta^L}$$

Retro-propagar el error a la capa anterior

$$\boldsymbol{\delta^{L-1}} = \boldsymbol{\delta^L w^L} \frac{\partial a^{L-1}}{\partial z^{L-1}} \tag{1.28}$$

Derivada parcial de cada capa usando el error

$$\frac{\partial C}{\partial w^{L-1}} = \boldsymbol{\delta^{L-1}} a_j^{L-2} \tag{1.29}$$

$$\frac{\partial C}{\partial b^{L-1}} = \boldsymbol{\delta^{L-1}} \tag{1.30}$$

1.3.2 Algoritmos de Optimización

Son metodologías que ayudan a minimiza o maximizar una función objetivo, por ejemplo, error o precisión de un modelo para así actualizar los pesos y polarizaciones de forma óptima con el menor número de pérdidas posibles. Trabaja en conjunto con los algoritmos de retro-propagación y gradiente descendente reciben los gradientes del primero, modificando el segundo para actualizar los pesos de manera adecuada [23].

1.3.3 Tipos de Optimizadores

1º Orden: Afectan a la función de pérdida usando su gradiente, siendo el más popular el gradiente descendiente, la derivada indica si la función incrementa o decrementa en un punto determinado mediante la línea tangente a la superficie del error, se caracteriza por un gasto computacional menor [23].

2º Orden: Usan derivadas de segundo orden o un arreglo de Hessian que es una matriz de derivadas parciales de segundo orden. Requieren un gran costo computacional, pero son capaces de indicar si la derivada de primer orden aumenta o disminuye mostrando la curvatura de la función, suministrando una superficie cuadrática que toca la función de error, estos requieren un mayor gasto computacional [23].

1.3.4 Adam

Adaptative Moment Estimation- *Estimación adaptativa del momento,* funciona con momentos de primero y segundo orden. Desarrollado bajo la premisa que no se quiere ir tan rápido para no pasar el mínimo, disminuyendo la velocidad para buscar cuidadosamente. Además, se tomar el promedio de disminución exponencial del cuadrado de los gradientes pasados como en AdaDelta. Llamado M(t) [23].

M(t) y V(t) son valores del momento de primer orden el cual es el promedio ec. (1.31) y el segundo es la varianza descentralizada de los gradientes respectivamente ec. (1.32):

$$\widehat{m}_t = \frac{m_t}{1-\beta_1^t} \tag{1.31}$$

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t} \tag{1.32}$$

Aquí, se toma el promedio de M(t) y V(t) así E[m(t)] puede ser igual a E[g(t)] donde E[f(x)] es un valor esperado de f(x).

Actualizando los parámetros mediante (1.33),

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} . m_t \quad (1.33)$$

Donde,

$\beta_1 = 0.9$, $\beta_2 = 0.9999$ $\epsilon = 10^{-8}$

1.3.5 Problemas con Entrenamiento de Red Neuronal

A pesar de todo existen problemas a la hora de realizar un entrenamiento como como. Subentrenamiento – *Underfitting* y Sobreentrenamiento-*Overfitting*

1.3.5.1 Subentrenamiento

Se refiere a que un modelo no es capaz de aprender de manera adecuada la representación de los patrones para así hacer buenas predicciones. Esto suele pasar porque la arquitectura es muy pequeña para capturar la complejidad del sistema o quizás este modelo no ha sido entrenado con suficiente data [13].

1.3.5.2 Sobreentramiento

En este caso el modelo ha aprendido bien, siendo capaz de predecir exactamente la data de entrenamiento, pero es incapaz de generalizar el aprendizaje con datos que no ha visto previamente, se dice que el modelo solo *memoriza* la data de entrenamiento o que aprendió a corto plazo, pero no la información del mundo real [13].

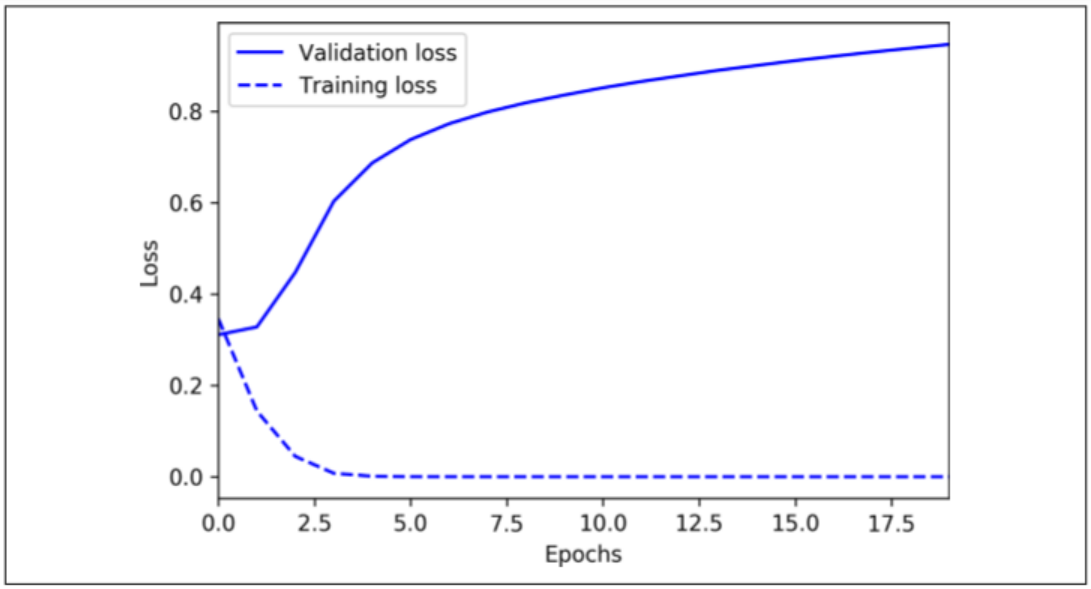


Figura 1.12 Gráfica pérdida de red neuronal con sobreentrenamiento *[13]*

1.4 Modelo de Regresión

Es un modelo que tiene como objetivo crear un modelo para poder predecir el valor de la clase datos a los atributos, esta usa clases continuas. Se usa por ejemplo árboles de regresión, regresión lineal, redes neuronales, LWR, etc [24]

1.5 Compressive sensing

La técnica Compressive Sensing disminuye las muestras necesarias para reconstruir una señal, reduciendo la frecuencia a la cual se debe tomar las muestras para una representación.

Su premisa es que una señal poco densa o Sparse es muy probable que se puede reconstruir a partir de un número reducido de muestras provenientes de su proyección aleatoria, siempre que la señal cumpla con la condición de escasez en algún dominio que se denomina diccionario ψ y que este a su vez sea incoherente con la matriz de medición Φ [25].

En Compressive Sensing (CS) se establece un número de mediciones M determinadas por el nivel de sparcidad de la señal, es decir, los valores diferentes de cero presentes en la misma, el vector de muestras ***y*** de una señal ***x*** tiene una longitud M, menor a las mediciones necesarias en el teorema de Nyquist.

El menor valor de M es el doble los valores k-Sparse de la señal, teniendo M>K<< N donde N es la longitud de la señal Sparse, se garantiza una reconstrucción apropiada con M > 3K, se realizan mediciones con un sub-

muestreo de pequeños conjuntos de valores de la señal nativa ***x.*** Teniendo en cuenta que ***x Є*** $\Re_n$ se afirma que cualquier vector perteneciente a este espacio puede ser representados en función de una base N *x* 1 vectores ψ_i con *i=1,2…N.* Por tanto, ***x*** es un vector que puede reconstruirse a partir de combinaciones lineales de la matriz ψ de longitud N x N y un vector ***s*** con valores significativos o distintos de cero de ***x*** como se observa en la **Figura *1.13* Representación del vector x en términos de ψ y s** dicho gráfico describe la ecuación (1.34):

$$x = \psi s \qquad (1.34)$$

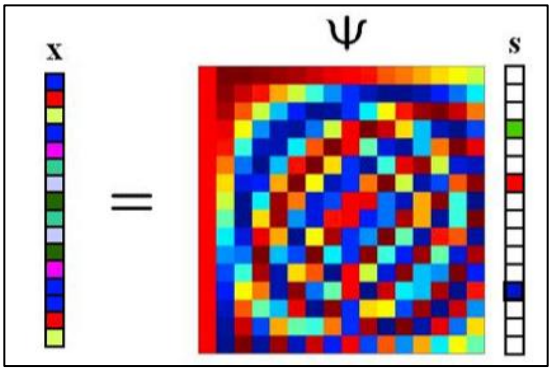


Figura 1.13 Representación del vector x en términos de ψ y s *[25]*

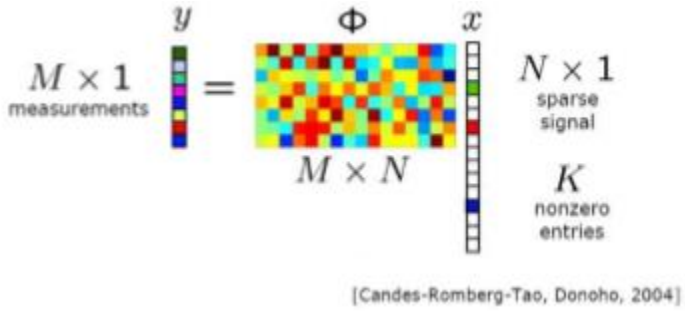
Entonces los valores s_i con i = 1, 2, …N se puede obtener a partir del producto $\psi_i \tau x$, donde la T indica transposición. Poniendo en evidencia que una señal Sparse puede ser reconstruida con valores significativos, ***x*** se muestrea adquiriendo una representación comprimida de la señal sin tomar N mediciones. Aquí es donde esta técnica cobra importancia con un muestreo a compresivo. El resultado de las operaciones se almacena en el vector de mediciones ***y*** con ayuda de la matriz aleatoria **φ**.

Las muestras son tomadas mediante productores entre la señal ***x*** y un arreglo de vector Φ_i que resulta en y_i , **Φ** también es conocida como la matriz de codificación, la cual está hecha para facilitar la reconstrucción de una señal Sparse, de dimensiones M x N, por tanto ***y*** se define como se ve en la ec. (1.35):

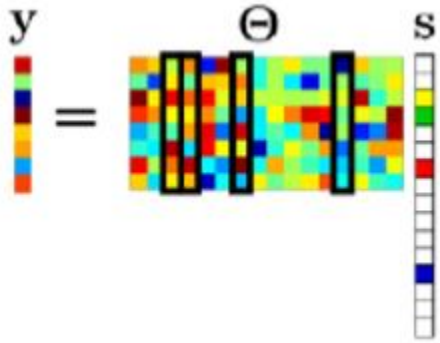
$$y = \Phi x = \Phi \psi s = \Theta s \qquad (1.35)$$

Donde en la ecuación(1.35) se define ***y*** de acuerdo a la previa representación de ***x,*** expuesta en la ecuación 1 , la matriz Θ de dimensiones N x M logra reducir las dimensiones al ser usada con ***s*** y es aleatoria. Contenidos en ***y*** se encuentran implícitos como productos de combinaciones lineales los valores K-

Sparse de ***x.*** El grado de compresión durante el muestreo es dado por el usuario y está determinado por el valor de M, que a su vez son las filas de la matriz aleatoria, las distintas pruebas sobre la señal permiten encontrar las mediciones óptimas entre nivel de compresión y una apropiada reconstrucción, como se ve en la **Figura *1.14* Muestreo en Compressive Sensing**.



(a) Producto de Φ_i por el vector columna X.



(b) Tomando la matriz aleatoria, se resaltan las columanas correspondientes a los valores significativos de S

Figura 1.14 Muestreo en Compressive Sensing *[25]*

1.5.1 Transformada Wavelet

1.5.1.1 Transformada Continua de Wavelet

La transformada wavelet de una función $f\langle t\rangle$ es la descomposición de la misma en un conjunto de funciones $\psi_{s,t}(t)$ que forman una base y son llamadas wavelets madres así enseña la ec. (1.36) [26]

$$W_f(s,\tau) = \int f(t)\,\psi^*_{s,\tau}(t)dt \tag{1.36}$$

Las Wavelets madres son generadas a partir de la traslación dado por el factor *s* y cambio de escala dado por τ contenido en (1.37)

$$\psi_{s,\tau}(t) = \frac{1}{\sqrt{s}}\psi\left(\frac{t-\tau}{s}\right) \quad (1.37)$$

Las funciones $\psi_{s,\tau}(t)$ generadas tienen distinta escala y ubicación, pero con la misma forma de onda. El factor de escala *s* siempre es mayor a cero, y son contraídas para s<1 y dilatadas para s>1, alterando el rango de frecuencias a tomar. Los mayores valores de s corresponden a un menor rango de frecuencia $(escala\ grande)$ mientras que los pequeños a un mayor rango $(escala\ pequeña)$

Dado lo anterior se afirma que las transformaciones Wavelets es una técnica **Multiresolución** ya que permite analizar señales en múltiples bandas de frecuencia. Tomados como una secuencia de subespacios cerrados [26] .

Cuando la resolución aumenta la función se aproxima a la original, mientras que al tender a cero se desprecia más información.

1.5.1.2 Transformada Discreta de Wavelet

La transformada continua de Wavelet presenta grandes complejidades debido a la variabilidad continua de los parámetros de escala como traslación, ha sido necesario desarrollar una herramienta de discretización. Tomando un conjunto finito de valores sustituyendo la integral por una sumatoria, y así representar la señal en términos de funciones elementales acompañadas de coeficientes [27].

Estos sistemas Wavelet traen consigo unas funciones de escala $\phi(t)$,las primeras se encargan de representar los detalles finos de la función, mientras las funciones de escala realizan una aproximación. Es posible representar la señal como una sumatoria de funciones wavelet y funciones de escala tal como indica (1.38).

$$f\langle t\rangle = \sum_k \sum_j C_{j,k}\phi(t) + \sum_k \sum_j d_{j,k}\psi(t) \quad (1.38)$$

Considerando una función discreta f_n

Se definen dos coeficientes A_n llamado de aproximación (1.39) y D_n llamado coeficiente de detalle (1.40):

$$A_n = \frac{F_{2n-1} + F_{2n}}{\sqrt{2}} \tag{1.41}$$

$$D_n = \frac{F_{2n-1} - F_{2n}}{\sqrt{2}} \tag{1.42}$$

La señal objetivo se descompone como se muestra en la **Figura *1.15*** [28] separando los componentes de baja frecuencia de las altas frecuencias obteniendo dos señales que producen el doble de muestras que la señal origina.

Por tanto, requiere disminuir el número de muestras a la mitad lo que se conoce como downsampling donde se involucran los coeficientes wavelets con una cantidad muestras reducidas a la mitad con respecto a los casos anteriores.

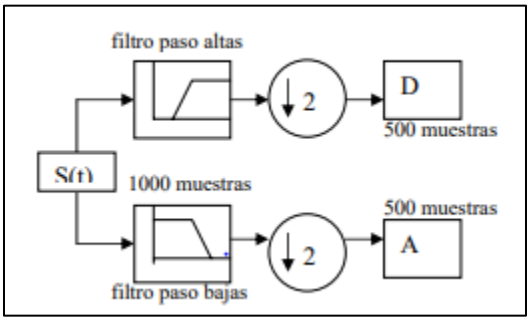


Figura 1.15 Descomposición y dowsampling de una señal con DWT [28].

1.5.2 Transformada de Fourier

Dada una función $f: \mathbb{R} \to \mathbb{C}$ se llamará transformada de Fourier de f a la función compleja definida por la ecuación (1.43) [29]:

$$\mathcal{F}|f(t)|(z) = \int_{-\infty}^{+\infty} f(t)e^{-izt}\,dt \tag{1.43}$$

Para todo $z\ \epsilon\mathbb{R}$ donde la expresión anterior tiene sentido si la integral impropia es convergente. Este hecho es más difícil de verificar que en la transformada de Laplace. Y debe cumplir lo expuesto en la ec. (1.44).

$$\lim_{t\to\pm\infty}\left|f(t)e^{-izt}\right| = \lim_{t\to\pm\infty}|f(t)| = 0 \quad (1.44)$$

Como en el caso de transformada de Wavelet el hecho de ser continua genera gran complejidad por lo anterior se hace necesario recuperar la señal a partir de distintas muestras generando la transformada discreta de Fourier DFT dada por la expresión (1.45)

$$X(k) = \sum_{n=0}^{N-1} x(n)e^{\frac{-j.2\pi kn}{N}} \quad (1.45)$$

1.5.3 Transformada Discreta del Coseno

Transformación numérica ampliamente usada en compresión de imágenes, esta es capaz de compactar la energía apropiadamente. La decorrelación de los coeficientes es importante para lograr una compresión adecuada ya que cada uno puede ser tratado de forma independiente sin perder eficiencia de la compresión. Dada por (1.46) [30]:

$$X(k) = \alpha(k)\sum_{n=0}^{N-1} x(n)cos(\frac{(2n+1)k\pi}{2N}) \quad (1.46)$$

1.5.4 Tensorflow

Es una librería desarrollada por Google Brain para sus aplicaciones de aprendizaje automático y redes neuronales profundas, la cual ejecuta de forma rápida y eficiente ***gráficos de flujo*** lo cual está conformado por operaciones matemáticas representadas sobre nodos cuya entrada y salida con vectores o tensores de datos [31].

1.5.5 Keras

Esta es una librería de redes neuronales escrita en Python, siendo una ilustración de una API de alto nivel para la creación de modelos de aprendizaje, aportando una sintaxis homogénea e interface sencilla, modular y ampliable para crear redes neuronales [32].

Las redes neuronales son un tipo de gráficos de flujo de datos por lo tanto keras y tensorflow combinan a la perfección para así potenciar la sencillez de uso y rapidez de ejecución.

1.5.6 API

Siendo en español interfaz de programación de aplicaciones, es un conjunto de protocolos que se usan para integrar y desarrollar software de aplicaciones, estas permiten que productos y servicios de un sitio se comuniquen con otros, sin saber cómo estos mismos se han implementado. Simplificando en tiempo y costo el desarrollo de aplicaciones, la estructura general de una API está dada por la **Figura *1.16* Arquitectura de una APIFigura *1.16*** [33] :

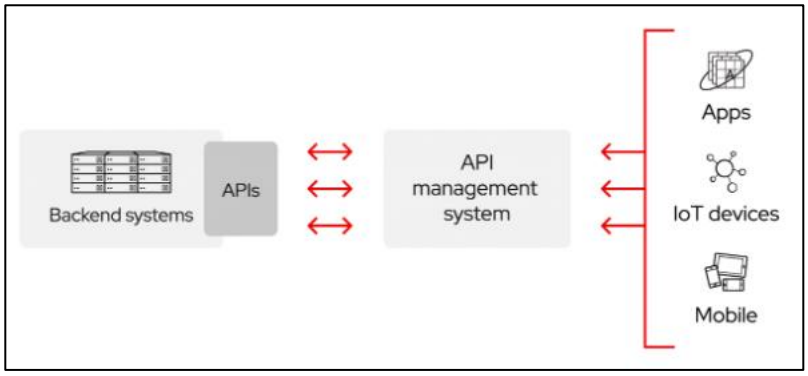


Figura 1.16 Arquitectura de una API *[33]*

1.5.6.1 Estilo REST

La mayoría de las compañías hacen uso de API REST para crear servicios. Donde el término REST se refiere a un conjunto de restricción con los que es posible crear una arquitectura de software, con lo cual es posible diseñar sitios web respetando HTTP [34].

Las restricciones de un sistema **RESTful** es:

1. **Cliente-Servidor:** Mantiene al cliente y al servidor débilmente acoplados. El primero no necesita conocer los detalles de implementación del servidor y este a su vez no tiene interés en el uso que da el cliente a los datos.
2. **Sin estado:** Cada petición que recibe el servidor deber ser independiente, no es necesario mantener sesiones
3. **Cacheable:** Debe admitir un sistema de almacenamiento en caché. La infraestructura de red debe soportar un caché de varios niveles. Para evitar repetir varias conexiones entre clientes y servidor para recuperar el mismo recurso.
4. **Interfaz Uniforme:** define una interfaz genérica para administrar cada interacción que se produzca entre cliente y servidor de manera uniforme,

lo cual simplifica y separa la arquitectura. Esta restricción indica que cada recurso del servicio REST debe tener una dirección URL.

5. **Sistema de capas:** el servidor puede disponer de varias capas para su implementación. Esto ayuda a mejorar la escalabilidad, rendimiento y seguridad.

Una API REST cuenta con 4 recursos a manipular, GET para consultar y leer, POST para crear, PUT para editar y DELETE para eliminar.

Además, es posible usar hipermedios, definido como los procedimientos para crear contenido que tengan texto, imagen, video, audio etc. Para así permitir que el usuario obtenga distintos recursos de la API mediante enlaces HTML.

Independientemente del lenguaje de programación usado la respuesta siempre se da con formato XML o JSON, ya que es el lenguaje de intercambio de información.

1.5.6.2 POWER Nasa

Es la base de datos abierta de datos meteorológicos de la NASA

Para consumir la API de este sitio buen se debe tener en cuenta que los datos medidos estarán disponibles 48 horas después de su fecha original , existe tres tipos de información

Medición en Punto Único- **Sample Single Point**, esta ofrece los datos de una ubicación geográfico específica se debe suministrar las coordenadas de latitud y longitud del lugar-

Medición Regional-**Sample Regional**, esta es una extensión del anterior permite crear una caja o figura cerrada con 4 puntos correspondientes a la zona a medir

Medición Global-**Sample Global**, es el último nivel de jerarquía y ofrece el estado a nivel mundial de un parámetro específico.

Para realizar la solicitud o requests se usa la url base de la suministrada por el sitio:

URL Base = https://power.larc.nasa.gov/cgi-bin/v1/DataAccess.py?

Figura 1.17 Url Base *[35]*

Solicitud: seguido a esto se debe indicar que desea realizar una solicitud o un request adiciona la url **https://...? request=execute** tal como ilustra a

```
URL=https://power.larc.nasa.gov/cgi-bin/v1/DataAccess.py? request=execute
```

Figura 1.18 URL Base indicando Request *[35]*

Identificadores: este parámetro indica el rango de medición se usa **SinglePoint**- para medir en un punto único, **Regiona**l- para una medición en la zona descrita por una caja. –**Global-** Para una medición mundial

Adicionando un indicador de área con las líneas https: **://...? identifier=SinglePoint** tal como indica la **Figura *1.19***

```
URL=https://power.larc.nasa.gov/cgi-bin/v1/DataAccess.py?
request=execute& identifier=SinglePoint& parameters=T2M,RH2M
```

Figura 1.19 URL con identificador de área *[35]*

Parámetros: Se debe especificar los parámetros a extraer, si se desea obtener más de un parámetro se debe separar por comas. Las mediciones en punto único y regional permiten como máximo 20 parámetros, mientras que las mundiales se limitan 3.

Cada parámetro se codifica con una palabra tal como indica la **Tabla *1.1***.

Tabla 1.1 Parámetros de la Base de datos *[35]*

Parámetro	Comando
Temperatura a 2m	T2M
Humedad Relativa 2m	RH2M
Velocidad del Viento a 10m	WS10M
Precipitación	PRECTOT
Insolación Horizontal	ALLSKY_SFC_SW_DWN

Por lo tanto se anexa la línea **https://...?parameters=T2M,RH2M** tal como indica la **Figura *1.20***

```
URL=https://power.larc.nasa.gov/cgi-bin/v1/DataAccess.py?
request=execute& identifier=SinglePoint& parameters=T2M,RH2M
```

Figura 1.20 URL con parámetros a obtener *[35]*

Fecha de Inicio y Fin: Se determinar el espacio temporal de los datos solicitados.

Tabla 1.2 Especificaciones rango de fechas *[35]*

Parámetros	Formato	Requisito para	Idenfiicador
startDate - endDate	YYYYMMDD	Diario	Punto único, Regional
startDate – endDate	YYYY	Interanual	Punto único, Regional
startDate - endDate	ND	Climatológico	Punto único, Regional, Global

Modificando la dirección como se ve en la **Figura *1.21*** restructurándola con la línea **https://...?startDate=YYYYMMDD&endDate=YYYYMMDD**

URL=https://power.larc.nasa.gov/cgi-bin/v1/DataAccess.py?
request=execute& identifier=SinglePoint& parameters=T2M,RH2M&
startDate=20200101&endDate=20201022

Figura 1.21 URL con Especificaciones rango de fechas *[35]*

Comunidad de Usuario: Este afectará las unidades de los parámetros y la forma de mostrar la serie de tiempo, se debe especificar una dirección tal como se observa en la **Figura *1.22*** donde se adiciona https://...?**userCommunity=SSE**

Tabla 1.3 Comunidades de usuarios *[35]*

Valor	Nombre
SSE	Superficie meteorológica y energía solar
SB	Edificios sostenibles
AG	Agroclimatología

URL=https://power.larc.nasa.gov/cgi-bin/v1/DataAccess.py?
request=execute& identifier=SinglePoint& parameters=T2M,RH2M&
startDate=20200101&endDate=20201022n& userCommunity=SSE

Figura 1.22 URL con indicador de Comunidad *[35]*

Promedio de Tiempo: determina la periodicidad de los datos solicitados, las opciones disponibles se ven en la ***Tabla 1.4*** agregando https://...?**tempAverage=DAILY** para completar una URL como en la ***Figura 1.23***

Tabla 1.4 Periodicidad de datos POWER NASA *[35]*

Parámetro	Valor	Identificador
tempAverage	DAILY, INTERANNUAL, CLIMATOLOGY	Punto único
tempAverage	DAILY, INTERANNUAL, CLIMATOLOGY	Regional
tempAverage	CLIMATOLOGY	Global

URL=https://power.larc.nasa.gov/cgi-bin/v1/DataAccess.py?
request=execute& identifier=SinglePoint& parameters=T2M,RH2M&
startDate=20200101&endDate=20201022n&
userCommunity=SSE& tempAverage=DAILY

Figura 1.23 URL con periocidad de datos *[35]*

Formato de Salida: Indica el formato de los datos en la salida por defecto el tipo JSON. Pero de igual forma se puede seleccionar CSV, ASCII, ICASA, NETCDF, bajo dos excepciones el formato ICASA no están disponibles para una selección de interanual o climatológica de datos regionales, ni para el tipo Global.

Se debe anexar, https://...?**outputList=JSON,ASCII** a la URL para que sea de acuerdo a la **Figura *1.24***

URL=https://power.larc.nasa.gov/cgi-bin/v1/DataAccess.py?
request=execute& identifier=SinglePoint& parameters=T2M,RH2M&
startDate=20200101&endDate=20201022n&
userCommunity=SSE& tempAverage=DAILY&
outputList=JSON,ASCII

Figura 1.24 URL con el Tipo de Salida del Archivo *[35]*

Coordenadas de Latitud y Longitud: Este parámetro determina la ubicación espacial de las mediciones, ya sea como un punto dado por la longitud y la latitud o un recuadro indicado por una zona, Se específica el sitio de las mediciones, anexando https://...?**lat=34.21&lon=56.78**

Tabla 1.5 Parámetros de Ubicación *[35]*

Parámetros	Formato	Nombre	Identificador
Lat	Grados decimales	Latitud	Punto único

Long	Grados decimales	Longitud	Punto único
Bbox	Latitud menor a la izquerda, longitud menor izquerda, Latitud mayor la derecha, Longitud mayor a la derecha	BoundingBox	Regional

La URL luego de aplicar esta modificación está dada por la **Figura *1.25***

URL=https://power.larc.nasa.gov/cgi-bin/v1/DataAccess.py?
request=execute& identifier=SinglePoint& parameters=T2M,RH2M&
startDate=20200101&endDate=20201022n&
userCommunity=SSE& tempAverage=DAILY& outputList=JSON&
lat=34.21&lon=56.78

Figura 1.25 URL con Ubicación Espacial *[35]*

Comunidad de Usuario: Esta en caso de que se desee colocar la organización que solicita los datos, no debe contener espacios, aunque usualmente tiene el valor de anonymous. (https://...?**user=anonymous**

)Se debe modificar la URL análogamente a lo realizado en las anteriores parametrizaciones finalizando con lo ilustrado en la **Figura *1.26***

URL=https://power.larc.nasa.gov/cgi-bin/v1/DataAccess.py? request=execute&
identifier=SinglePoint& parameters=T2M,RH2M&
startDate=20200101&endDate=20200102n& userCommunity=SSE&
tempAverage=DAILY& outputList=JSON
&lat=34.21&lon=56.78&user=anonymous

Figura 1.26 URL con Comunidad de Usuario *[35]*

Finalmente, esa es la URL (**Figura *1.26***) necesaria para realizar una solicitud al sitio POWER NASA.

CAPÍTULO II

Desarrollo Metodológico

La estructura general del desarrollo metodológico se observa en la **Figura *2.1*** donde se registra el dataset necesario desde la base de datos POWER NASA que a pesar de todo tiene valores erróneos por lo que fue necesario aplicar un algoritmo de reconstrucción con vecinos cercanos, posteriormente se generan las características para un adecuado entrenamiento donde se divide el vector en series de tiempo individuales, se convierte a Sparse para posteriormente aplicar Compressive Sensing técnica de extracción de patrones y reducción de longitud, teniendo todo preparado para crear un modelo y posteriormente entrenarlo y validarlo.

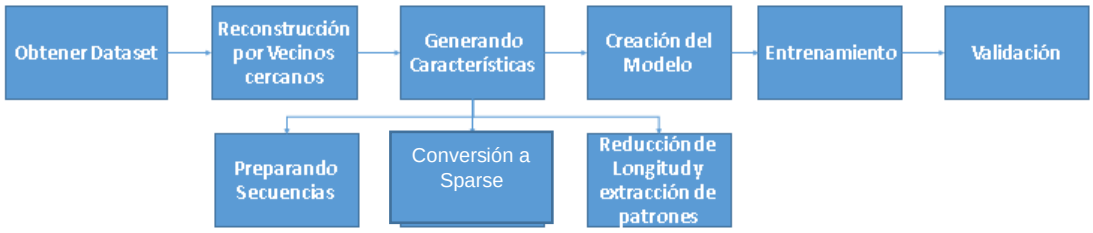


Figura 2.1 Metodología de entrenamiento de datos [Elaboración Propia]

2.1 Registro del Dataset

La arquitectura de Deep Learning podría ser capaz de pasar por alto la información irrelevante, pero se recomienda un entrenamiento con únicamente patrones distintivos para resolver de forma adecuada el problema. Así la arquitectura anulará entradas irrelevantes, siendo incapaz de aprender información falsa.

Reconocer la data importante implica evaluar la complejidad de las relaciones entre variables, teniendo en cuenta la regla de oro "***entre más data mejor".*** En este aspecto la institución de servicio climatológico colombiano **IDEAM** en muchas estaciones no tenía completamente los datos necesarios, además de mediciones erróneas, sumado a la poca información de una API, por lo tanto, se determinó usar una plataforma internacional eligiendo la base de datos meteorológicos **NASA Power** la cual cuenta con una API activa y pública, Además de un instructivo claro de cómo consumir la misma, con mediciones a partir de 2015. Se tomaron dos ubicaciones determinadas por latitud y longitud correspondientes a (4.355, -74.226) y (16.5337, -80.2263) seleccionando los promedios diarios de variables como **Temperatura**, **Humedad Relativa**, **Velocidad del Viento**, **Precipitación**, ya que estos cuatro están categorizados como los principales elementos del clima y permiten comprender el estado del

tiempo de un lugar, además de **Insolación** que corresponde a una relación inversa en este caso con la nubosidad presente. Como se ilustra en la ***Tabla 2.1*** junto con las etiquetas referenciadas por la base de datos.

Tabla 2.1 Variables Tomadas

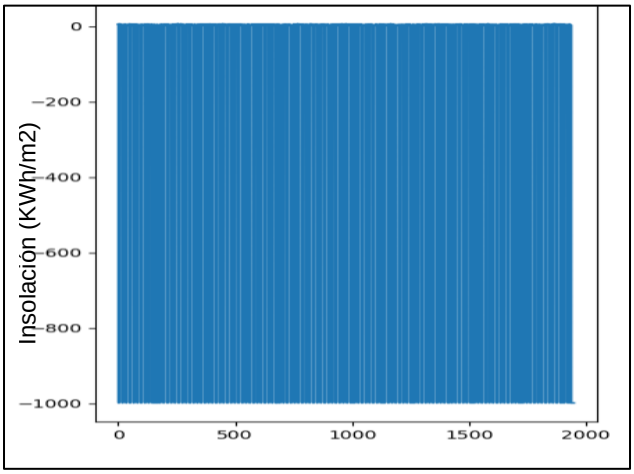
Parámetro	Comando
Temperatura a 2m [ºC]	T2M
Humedad Relativa 2m [%]	RH2M
Velocidad del Viento a 10m [m/s]	WS10M
Precipitación [mm/día]	PRECTOT
Insolación Horizontal [Kwh/m_2]	ALLSKY_SFC_SW_DWN

2.2 Reconstrucción con vecinos cercanos

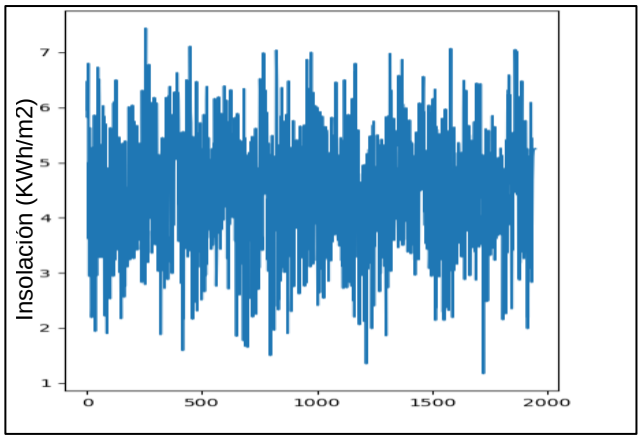
Los datos registrados en algunas variables tal como la insolación presentaban información faltante, el proveedor de los mismos reemplaza los valores por la cifra **-999.0** tal como se ve en la **Figura *2.3*** (a), esto presenta un inconveniente en el procesamiento y aprendizaje de patrones del modelo a diseñar, por tanto, se desarrolló un algoritmo basado en vecinos cercanos para intentar reconstruir la información ausente una gráfica como las de la **Figura *2.3*** (b), el fundamento de dicho algoritmo se basa en identificar una medición errónea y promediar el dato **anterior** con el próximo dato **definido**, es decir, **diferente de -999.0,** no necesariamente debe ser el dato de la posición siguiente, se tuvo en cuenta cuando habían 2 o más datos adyacentes no validos para omitirlos hasta el próximo parámetro valido, en el desarrollo del mismo se observaron dos casos especiales y es cuando el primero o el último dato estaban equivocados ya que se hacía imposible obtener el datos anterior o el siguiente, entonces la primera condición es asegurarse que estos datos sean distintos de -999.0 reemplazándolos por datos aleatorios del vector y así posteriormente reconstruir los datos en su totalidad, bajo las premisas condensadas en la **Figura *2.2***

```
Mientras el último o el primer dato sean iguales -999.0
        Si el último o el primer dato son -999.0
                Se reemplaza por un dato aleatorio
Para todos los datos del vector
        Si un dato es igual a -999.0
                Tomo el dato de la posición anterior
                Busco el próximo que sea distinto a -999.0
                        Promedio ambos datos
                        Lo ubico en la posición del dato faltante
```

Figura 2.2 Metodología de Reconstrucción de datos [Elaboración Propia]



(a) Señal con datos faltantes



(b) Señal Reconstruida con algoritmo de vecinos cercanos

Figura 2.3 Algoritmo de Reconstrucción por vecinos cercanos [Elaboración Propia]

2.3 Generando características de los datos

Se sabe que los modelos necesitan tensores, pero en la mayoría de los casos es necesario observar cómo transformar esa información para que sea significativa. En ML el término característica se refiere a una información particular con la cual es entrenado el modelo. Las entradas pueden ser desde

valores escalares simples hasta complejos arreglos de imágenes que encaja en tensores multidimensionales.

En el entrenamiento de las redes neuronales los datos deben tener pequeñas amplitudes por tanto, se aplica la conversión de la ecuación (2.1), tomando cada elemento del vector sustrayéndole la media del conjunto de valores y dividiéndolo entre la desviación estándar de los mismos, así se logra que la función de pérdida RMSE (error cuadrático medio definida por la ecuación (1.10)) sea inicialmente 1.0 y esté más cerca a llegar a cero (0.0) durante el entrenamiento .

$$X_{esc} = \frac{X - Media_X}{Desviación_X} \tag{2.1}$$

2.3.1 Preparación de Secuencias de tiempo

Para preparar las secuencias de tiempo cabe destacar que inicialmente se tuvo un vector de longitud 1712, este se dividió ventanas de 131 datos con avance de un posición, dónde las primeras 130 posiciones referente a los patrones se almacenan en las 1581 filas de la matriz **P_S,** a estos se les asignó como respuesta el parámetro adicional (posición 131) y se acomodó en la fila correspondiente del vector de etiquetas **E_S.**

De igual forma se implementó la **predicción múltiple**, este modo incluye el término llamado longitud de predicción (*lp*) que determina el número de columnas del ahora matriz de etiquetas **E_M**, no se ven afectadas las dimensiones de sus características (130), se tuvo en cuenta la variable *lp* para evitar un desbordamiento en la extracción de patrones. Esto ocasionó que las filas de **P_M** disminuyera en proporción de la longitud de predicción.

$$\mathrm{P\_S} = \begin{bmatrix} a_{11}^{1} & a_{12}^{2} & \cdots & a_{1R}^{130} \\ a_{21}^{2} & a_{22}^{3} & \cdots & a_{2R}^{131} \\ \vdots & \vdots & \ddots & \vdots \\ a_{M1}^{1649} & a_{M2}^{1650} & \cdots & a_{MR}^{1711} \end{bmatrix} \quad E\_S = \begin{bmatrix} a_{11}^{131} \\ a_{21}^{132} \\ \vdots \\ a_{M1}^{1712} \end{bmatrix}$$

P_S = Patrones Predicción Simple.
E_S = Etiquetas Predicción Simple

$$a_{ij}^{k}\ i = filas\ j = Columnas\ k = posición\ en\ el\ dataset\ R = 130\ M = 1581$$

$$P\_M = \begin{bmatrix} a_{11}^{1} & a_{12}^{2} & \cdots & a_{1R}^{130} \\ a_{21}^{2} & a_{22}^{3} & \cdots & a_{2R}^{131} \\ \vdots & \vdots & \ddots & \vdots \\ a_{M1}^{1648-lp} & a_{M2}^{1649-lp} & \cdots & a_{MR}^{1711-lp} \end{bmatrix} \quad E\_M = \begin{bmatrix} a_{11}^{131} & \cdots & a_{1R}^{131+lp} \\ a_{12}^{132} & \cdots & a_{1R}^{132+lp} \\ \vdots & \ddots & \vdots \\ a_{M1}^{1712-lp} & \cdots & a_{MR}^{1712} \end{bmatrix}$$

P_M = Patrones Predicción Múltiple.
E_M = Etiquetas Predicción Múltiple.

$a_{ij}^{k}\ i = filas\ j = Columnas\ k = posición\ en\ el\ dataset$

$lp = [2, 5{,}10]\ R = 130\ M = 1581 - lp + 1$

La **Figura *2.4*** enseña una serie de tiempo pudiendo ser cualquier fila de las matrices **P_M** o **E_M**

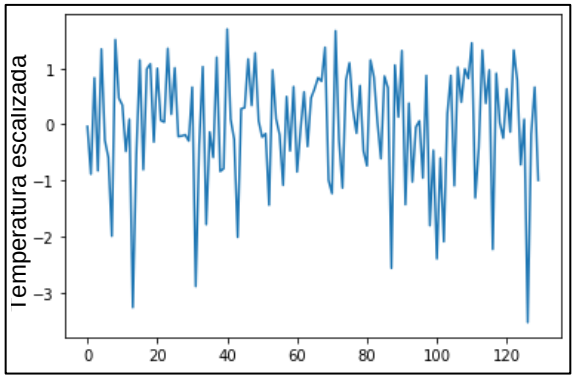


Figura 2.4 Serie de Tiempo Original (longitud: 130 datos). [Elaboración Propia]

2.3.2 Conversión a Sparse

De acuerdo a la teoría de CS esta menciona que esta metodología se aplica a señales en el espacio Sparse por tanto el primero paso es aplicar las transformaciones necesarias para lograr dicho objetivo.

Se realizó una comparación entre la transformada de kronecker aplicando la función delta de kronecker dada por la ecuación (2.2) y por la técnica compuesta de la **Figura *2.6***.

$$\delta_n = \begin{cases} 1\ Sí\ n = Umbral \\ 0\ Sí\ n \neq Umbral \end{cases} \qquad (2.2)$$

La primera forma definida por la ecuación (2.2) usó un umbral compuesto por 16 valores (Vector [16x1]) dentro de dataset de manera aleatoria como mediciones importantes dentro del arreglo como resultado se obtiene que esta es excepcional para convertir al espacio sparse arrojando respuestas como se ve en la ***Figura 2.5***

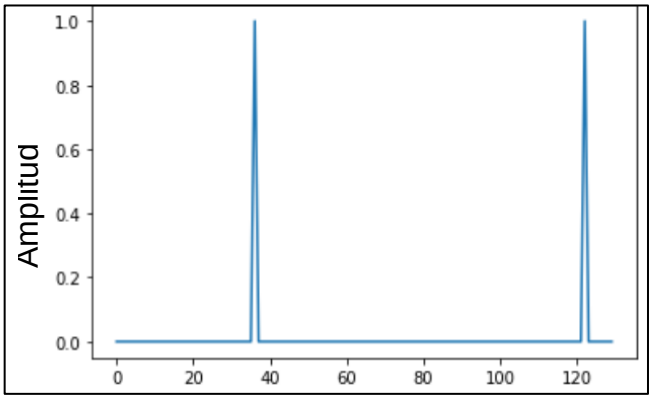


Figura 2.5 Conversión al espacio Sparse usando la función de Kronecker

En la técnica 2 el procesamiento se dividió en dos etapas contenidas en su totalidad en la **Figura *2.6*** donde se consideró aplicar ***transformada de Fourier*** tomar su parte real y a dicho conjunto de datos aplicar ***transformada de coseno*** para compactar toda información de cada secuencia en pocos puntos.

La segunda parte tomó distintas ventanas de datos para el caso en específico se eligieron grupos de 10 valores operando bajo de la instrucción ***wavedec*** comando que descompone la señal aplicar transformada wavelet al nivel indicado, concatenando el último coeficiente de aproximación con los coeficientes de detalles, entre mayor la descomposición se obtienen puntos cercanos a cero, tomaron 13 niveles ya que como se observa en la ***Tabla 2.2*** un valor mayor no correspondía a una disminución significativa de la sparsidad medida aplicando la norma L1. Se usó wavelet madre **Daubechies 1** ya que la gran de mayoría de esta familia convertían a apropiadamente a Sparse cada secuencia debido a que se estructuran condensando un impulso en su parte central, mientras que otras familias hacían transformaciones a formas de onda no – Sparse. En general esta última técnica es de gran relevancia en conseguir una sparsidad idónea, reflejado en la ***Tabla 2.3*** que encapsula la dispersión de las señales al omitir **wavedec** obteniendo valores de dispersión elevados con relación a los finalmente seleccionados para la Compresión de las Señales.

Tabla 2.2 Medición de Sparsidad Para Cada Parámetro con las técnicas de la Figura 2.6

Variable	Sparsidad Promedio					
	10 niveles		13 niveles		16 niveles	
	Ubicación		Ubicación		Ubicación	
	1	2	1	2	1	2
T2M	6.48	5.87	4.75	4.78	4.38	4.42
RH2M	6.51	6.17	4.79	4.63	4.12	4.28
WS10M	6.56	6.46	4.78	4.79	4.38	4.41
PRECTOT	6.00	6.59	4.50	4.41	4.16	4.06
ALLSKY_SFC_SW_DWN	6.35	6.48	4.69	4.75	4.33	4.37

Tabla 2.3 Medición de Sparsidad de Variables sin Aplicar Wavedec

Variable	Sparsidad Promedio	
	DCT-FFT	
	Ubicación	
	1	2
T2M	40.63	37.00
RH2M	40.61	37.03
WS10M	40.66	41.95
PRECTOT	57.63	43.87
ALLSKY_SFC_SW_DWN	43.37	43.50

Figura 2.6 Metodología Conversión a espacio Sparse [Elaboración Propia]

Posteriormente se escalizaron los datos a un rango de (-1,1) haciendo uso de la ecuación (2.3) la cual es análoga la utiliza en la función **map** de los microcontroladores de la familia Atmel bajo el entorno de programación de Arduino, ya que el procesamiento de la **Figura *2.6*** aumentaba en gran manera los niveles numéricos de la señal, a grandes rasgos esta busca identificar los puntos máximos y mínimo del vector de entrada tanto como los deseados y así redefinir cada valor.

$$x_{esc} = \frac{[x - \max(x)][1 - (-1)]}{\max(x) - \min(x)} - 1 \qquad (2.3)$$

Además, en algunas ocasiones las señales tenían un pequeño nivel de offset el cual no es óptimo para señales Sparse lo que llevo a desarrollar el algoritmo de la **Figura *2.7*** el cual consiste en identificar el nivel offset que corresponde a la región encerrada en un círculo rojo de la **Figura 2.8** que a su vez cumple con la condición de tener una derivada discreta igual a cero por lo tanto luego de reconocer este valor se procede a restarlo a cada valor de vector y así lograr que la mayoría de los datos sean iguales a cero y no a ese nivel no deseado

```
Para i=1 hasta longitud(x)-1
        Si x[i+1] – x[i] es igual a 0
                ∀x = ∀x − x[i]
                Salir
        Fin
Fin
Nota: ∀ -> Todos los valores de un conjunto
```

Figura 2.7 Algoritmo de corrección nivel de offset [Elaboración Propia]

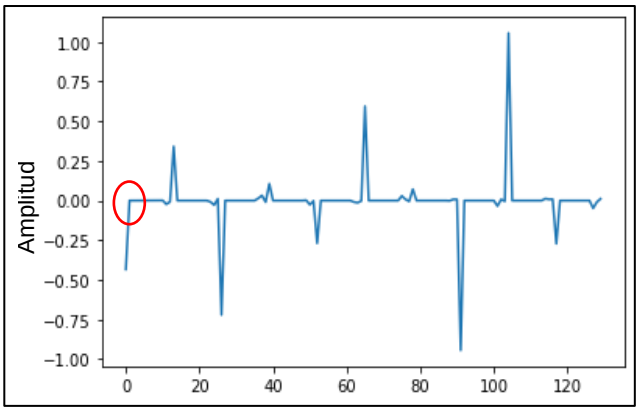


Figura 2.8 Serie de Tiempo en el espacio Sparse **[Elaboración Propia]**

Para elegir una sola línea de trabajo se optó por la técnica 2, ya que la función **delta de kronecker** fue apta para convertir al espacio sparse, pero tenía el inconveniente de poseer solo dos niveles de amplitudes esta **poca diversidad** en comparación a la segunda metodología representaba una pobre extracción de patrones.

2.3.3 Reducción Longitud de datos

Para la disminuir a longitud de la secuencia se crearon dos matrices una matriz identidad cuadrada, además de la transformada coseno inversa de la misma, con las dimensiones de una secuencia ***130 x130***, a continuación a esto se realiza un recorte del número de filas determinado por la ecuación (2.3) [Ecuación de Bernoulli] donde N = 130 dado por la columnas de matriz inicial, *k=5* es la sparsidad promedio tomada por la ***Tabla 2.2*** como el valor inmediatamente superior a la dispersión estimada para 13 niveles de descomposición con **wavedec** y C=2

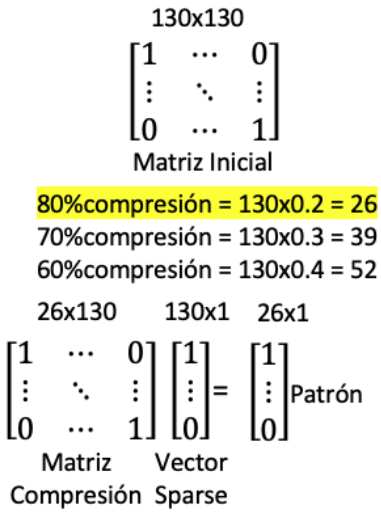
$$filas > C * \log(N/k) \qquad (2.4)$$

$$filas > C * \log\left(\frac{N}{k}\right) \qquad filas > 2 * \log\left(\frac{130}{5}\right) \qquad filas > 14$$

Por tanto, se toma un número de muestras correspondientes al 20%, 30% y 40% de las filas, siendo 26, 39 y 52, valores que cumplen la ecuación de **Bernoulli** y se relacionan directamente con compresiones del 80%, 70% y 60%, esto debido a que un 90% de condensación (muestras del 10%) descartaba información relevante mientras que compactaciones de un 50% o superior, aumentaba el tiempo de procesamiento sin ninguna mejora en la predicción. Dado lo anterior, obteniendo un vector de la longitud ***muestrasx130,*** posteriormente cada secuencia se toma como vector columna de **130x1** al multiplicar (***muestrasx130*** x **130x1)** la porción de la matriz se obtienen unas nuevas secuencia de longitud de ***muestrasx1*** el valor de muestras es igual a 130 por un valor llamado porcentaje de muestras llamado en el algoritmo de la **Figura *2.9*** **porc_muestras** que tiene valores 0.2, 0.3 y 0.4 dando como resultado tensores de longitudes redondeadas de 26, 39 y 52.

```
A = Matriz_de_Compresión %nxn
A = A [0: n*porc_muestras] %muestrasxn
Patron = AxSeñal_Sparce   % muestrasx1
```

(a) **Algoritmo de Aplicación de CS para reducción de Longitud**



(b) Descripción matricial CS

Figura 2.9 Extracción de patrones [Elaboración Propia]

n = 130
porc_muestras (porcentaje de muestras) = [0.2, 0.3 o 0.4]

La extracción de patrones con cada matriz se ve en la **Figura *2.10*** y **Figura *2.11***

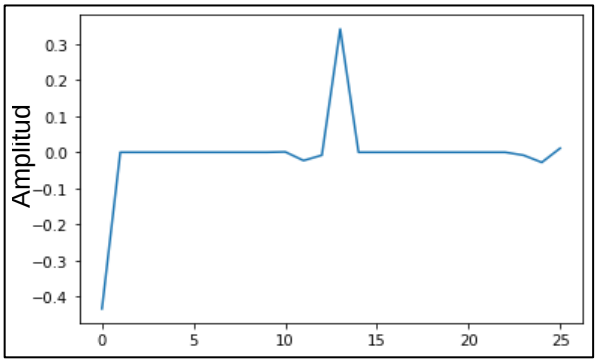


Figura 2.10 Aplicando Compressive Sensing con una Matriz identidad de 26x130 [Elaboración Propia]

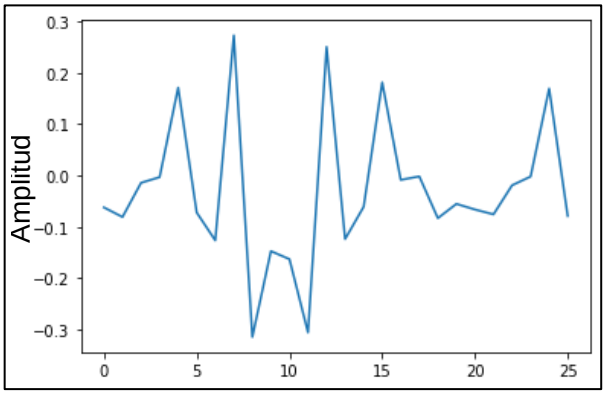


Figura 2.11 Aplicando Compressive Sensing con una Matriz identidad con transformada inversa de coseno de 26x130 [Elaboración Propia]

2.4 Diseñando Arquitectura del Modelo

Existen muchos diseños de aprendizaje profundo desarrollados para resolver un gran rango de problemas. Es posible seleccionar un modelo ya preconcebido, así como basarse en uno conocido. Este decide de acuerdo a l aplicación, el tipo de data a la cual se accede y las formas de transformaciones en el pre-procesamiento. Sin olvidar cualquier limitación que pueda tener el proyecto.

Al enfrentarse a un modelo de predicción es necesario conservar la misma escala de valores propias de cada variable por lo que la única función de activación óptima es la tipo ***linear*** usualmente se toma la dimensión de salida como 1 [uno], para así a partir de un patrón o secuencia característica de una serie de días anteriores, en este caso 130, y así predecir las condiciones correspondientes a las fechas siguiente, aunque se adicionaron pruebas con dimensiones de salida de 2, 5 y 10 (días).

Para las capas de entrada y ocultas se consideró el uso de funciones activación óptimas para redes neuronales multicapa alternando entre **ReLu** y Tangente

Hiperbólica (**tanh**) con un número respectivo de neuronas y capas que propiciaran un aprendizaje rápido en el menor tiempo posible.

Se seleccionó el optimizador ***Adam*** ya que es la versión mejorada de todas las técnicas siendo uno de los algoritmos más rápidos a la hora de actualizar los pesos y polarizaciones del modelo, además de tomar **Error Cuadrático Medio** (**RMSE**) como función de pérdida la cual es óptima para regresiones ya que miden que tan distante esta un dato predicho de uno real.

Las Arquitecturas que se implementaron son las siguientes tanto en **Tabla *2.4*** como en **Tabla 2.5**

Tabla 2.4 Arquitectura 1 [Elaboración Propia]

Arquitectura 1		
Número de Capas	8 capas	Matriz de Compresión
Optimizador	Adam	IDCT de Identidad
Capa	Función de Activación	Número de neuronas
Capa 1	Tanh	300
Capa 2-7	ReLu	400
Capa 8	Linear	400

Tabla 2.5 Arquitectura 2 **[Elaboración Propia]**

Arquitectura 2		
Número de Capas	8 capas	Matriz de Compresión
Optimizador	Adam	Identidad
Capa	Función de Activación	Número de neuronas
Capa 1	ReLu	300
Capa 2-7	ReLu	500
Capa 8	Linear	500

2.5 Entrenar el Modelo

El entrenamiento es un proceso por el cual el modelo aprende a generar las salidas correctas para unas entradas dadas. Esto implica suministrar datos y hacer unos pequeños ajustes hasta la máxima precisión posible. Las modificaciones se llevan a cabo en los arreglos que representan las neuronas conocidas en el ámbito científico como *weights*-Pesos y *biases*-polarización.

Se usó la instrucción tic-toc para poder medir el tiempo que demora en realizar el entrenamiento.

Además, se tomó un 10% para la validación del modelo, se verificó el error cuadrático medio y la precisión cada 100 épocas. Para la arquitectura 1 se usaron 100 épocas, mientras que para el segundo 200 debido a la lentitud de la misma para reducir la función de pérdida hasta cero.

Tabla 2.6 . Datos de entrenamiento

Datos de entrenamiento	Datos de validación	Datos de evaluación
1712	172	190

2.6 Evaluando el Modelo

Para evaluar el modelo se tomó la última secuencia de la matriz de entrenamientos a la cual se le aplicó el procesamiento de la **Figura *2.12***, este consiste en considerar los 130 valores anteriores (incluyendo el último) obteniendo la predicción correspondiente, a partir de este dato se toma una ventana considerando el o los dato ya predichos y se repite el procedimiento.

```
Para i Desde 0 a pasos de Lp hasta Tp
        data_pre = ultima_secuancias [i: l_s+i]
        data_pro = procesamiento[data_pre]
        Nuevo_dato = predicción[data_pro]
        Anexo [Nuevo_dato] a data_pre
```

Figura 2.12 Algoritmo de Evaluación del Modelo [Elaboración Propia]

Lp = Longitud de Predicción [2, 5 o 10]
Tp = Tiempo de predicción [190]
l_s = Longitud de secuencia [130]

Luego que los datos son calculados se debe revertir la normalización aplicada al principio de la sección 2.3 y se aplica la ecuación 2.3

$$X_{pre} = X_{esc} * Desviación_X + Media_X \quad (2.5)$$

$X_{pre} = datos\ predichos\ con\ normalización\ inversa$
$X_{esc} = datos\ predichos\ normalizados$

2.7 Plataforma de predicción del estado del tiempo

Para enseñar los modelos desarrollados en un sistema lo más parecido a la realidad se desarrolló la plataforma de predicción del estado del tiempo **ETHAN** la cual obtiene los datos de la API correspondiente al sitio **POWER NASA** para posteriormente aplicar el mismo procesamiento de las secciones anteriores desde la reconstrucción por vecinos cercanos pasando por la generación de características con su respectiva conversación a señal Sparse y extracción de patrones para aplicar un respectivo modelo de predicción y brindar distintas formas de visualización.

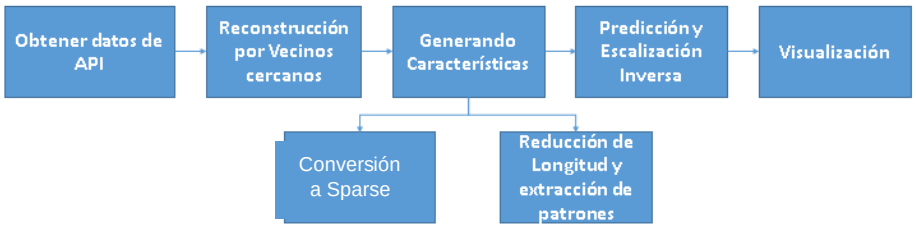


Figura 2.13 Funcionamiento de Plataforma ETHAN [Elaboración Propia].

Para realizar la solicitud o request de los datos se usó la API del sitio POWER NASA, usando el método **GET** usando como url base la suministrada por el sitio, más los datos específicos:

Tabla 2.7 Datos de la solicitud

Parámetro	Valor	
Identifier	SinglePoint	
Parameters	Variables TABLA X	
StartDate	132 días antes de la fecha actual	
endDate	2 días antes de la fecha actual	
userCommunity	UserCommunity	
tempAverage	DAILY	
Lat	4.355	16.5337
Lon	-74.226	-80.2263
User	Anonymous	

Formulando una URL como se observa en la **Figura *2.14***, se consideró el ejemplo de la temperatura diaria a 2m desde el 01/01/2017 hasta el 22/10/2020

```
URL=https://power.larc.nasa.gov/cgi-bin/v1/DataAccess.py? request=execute&
identifier=SinglePoint& parameters=T2M&
startDate=20170101&endDate=20201022n& userCommunity=SSE&
tempAverage=DAILY& outputList=CSV& lat=4.355&lon=-
74.226&user=anonymous
```

Figura 2.14 URL de ejemplo Usada para la solicitud

Una mayor información de la estructura de dicha URL se específica en el **Capítulo I – sección 1.4.6.2**

El primer paso es realizar una solicitud a la dirección anterior a su vez esta genera el diccionario en formato JSON de la **Figura *2.15*** con las siguientes secciones enfocándose en el diccionario **outputs**

Clave/Tecla	Tipo	Tamaño
features	list	1
header	dict	5
messages	list	0
outputs	dict	1
parameterInformation	dict	1
time	list	2
type	str	1

Figura 2.15 Diccionario JSON como respuesta de la Solicitud

El contenido de outputs condensado en la **Figura *2.16*** se compone de un string con la dirección URL de descarga del archivo CSV esta

Clave/Tec	Tipo	Tamaño	Valor
csv	str	1	https://power.larc.nasa.gov/downloads/POWER_SinglePoint_Daily_20191222 ...

Figura 2.16 String con dirección de descarga del archivo CSV

Luego se usa Pandas para así leer el archivo CSV de la dirección accediendo directamente a los datos, sin necesidad de descargarlos el resultado está en la **Figura *2.17***

prd - DataFrame

Índice	LAT	LON	YEAR	MO	DY	T2M
0	4.35571	-74.2263	2019	12	22	18.09
1	4.35571	-74.2263	2019	12	23	18.38
2	4.35571	-74.2263	2019	12	24	18.49
3	4.35571	-74.2263	2019	12	25	18.09
4	4.35571	-74.2263	2019	12	26	18.97
5	4.35571	-74.2263	2019	12	27	18.7

Figura 2.17 Datos leídos de la API

Luego de obtener los datos se aplica el siguiente algoritmo para realizar la predicción, teniendo los modelos y la medía tanto como la desviación estándar de los datos de entrenamiento para su respectiva normalización y reconstrucción, se aplica la secuencia de la **Figura *2.18*** el cual inicialmente toma los datos de la api, la media y desviación estándar específicos de cada parámetro específico para normalizarlos, posteriormente se envía una matriz base cuadrada de 130x130 ya sea identidad o esta misma luego de haber aplicado transformada discreta inversa de coseno, junto con un **porcentaje de muestras** para así obtener un tensor 2D de dimensiones ***muestrasx130 ,*** en la siguiente línea dicha matriz se usa junto con los datos normalizados para el vector de patrones, además de un modelo que permita predecir las condiciones de acuerdo a un conjunto de días llamados **lp,** aquí se hace uso nuevamente la medía y desviación para así revertir la normalización :

```
da_esc = normalizar [datos_api, media, desv_est]
Matriz= crearMatriz [datos_esc, Matriz_base, (1-%compresión)]
Datos_p=predicción [da_esc, matriz, modelo, media, desv_est, lp]
Datos  p = Datos  p [2: longitud (Datos  p)]
```

Figura 2.18 Algoritmo de Predicción en la interfaz ETHAN

da_esc = datos escalizados luego de aplicar la ec. (2.1)
Matriz_base = es la matriz inicial creada [identidad- dct(identidad)]
Datos_p= los datos predichos ya en su amplitud de entrada
Lp = Número de datos estimados

*Al final del algoritmo se desprecian los dos primeros datos ya que es el último dato del vector de entrada no es el actual sino 48 horas antes, esto determinado por la base datos usadas.

Se desarrollaron 2 modos de vista en la plataforma ETHAN donde el primero permite visualizar los datos como una gráfica a través del tiempo con un eje de

abscisas con las fechas correspondientes a cada valor como se ve en la **Figura *2.19***, los botones de flechas de la parte superior permiten avanzar en el tiempo

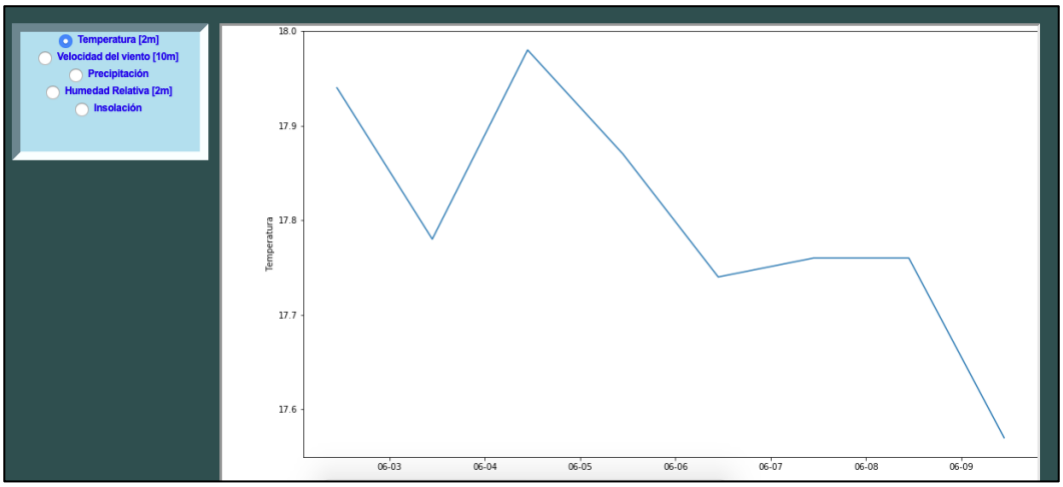


Figura 2.19 Vista Modo 1 plataforma ETHAN

Por otro lado en la **Figura *2.20*** se observa la segunda pestaña que encierran en distintos cuadros una fecha determinada con las condiciones para la misma los botones superiores se mantienen cumpliendo la misma funcionalidad.

Figura 2.20 Vista Modo 2 plataforma ETHAN

CAPÍTULO III

Resultados

A continuación, se presentan los resultados obtenidos en el desarrollo del proyecto.

3.1 Acondicionamiento de los datos

3.1.1 Normalización

El impacto real de aplicar la formula (2.1) se ilustra en la **Figura 3.2** donde los niveles de temperatura pasaron de estar entre [15 21] en la **Figura *3.1*** a estar en magnitudes entre [-3 4] las cuales son más fáciles de trabajar para cualquier modelo de entrenamiento.

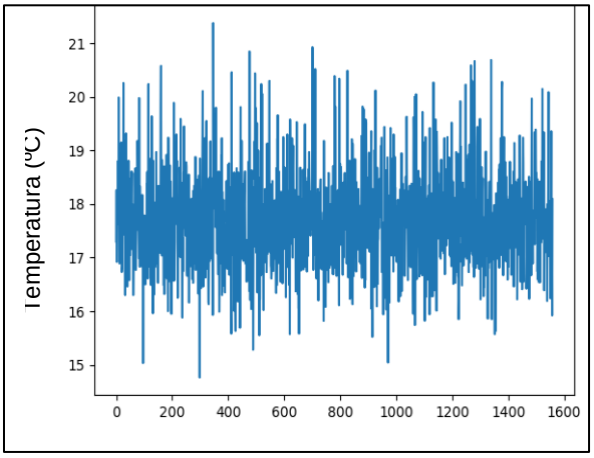


Figura 3.1 Historial de Temperatura no Normalizada

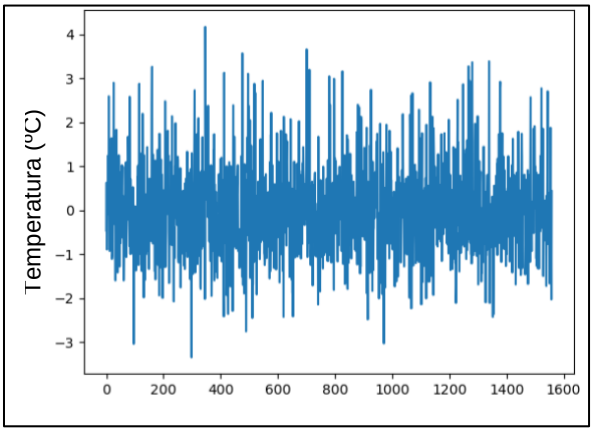


Figura 3.2 Historial de Temperatura Normalizado

En variables como la Velocidad del Viento no se notó mucho la normalización de los datos ya que estos en su naturaleza tienen magnitudes pequeñas por tanto los niveles de la **Figura *3.3*** no son tan distintos a los de la **Figura *3.4***

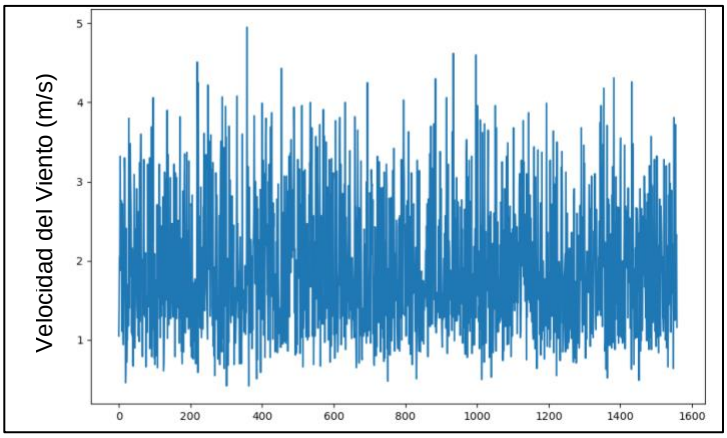


Figura 3.3 Velocidad del Viento No Normalizada

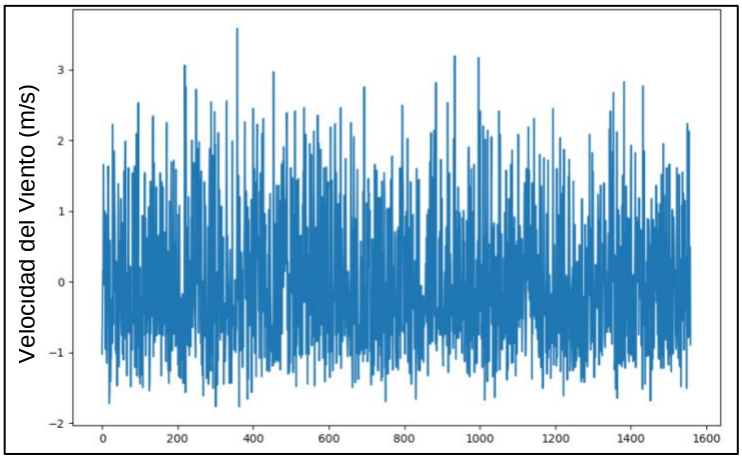


Figura 3.4 Velocidad del Viento Normalizada

3.1.2 Conversión a espacio Sparse

Al aplicar el algoritmo de la **Figura *2.6*** se convierten las series de tiempo en la forma de onda de la **Figura *3.5*** pero como se observa su nivel base no es cero tal como lo indica el círculo rojo por lo que al aplicar la secuencia de la **Figura *2.7*** se logra que la mayoría de los datos sean lo más cercanos a cero tal como se ilustra en la **Figura *3.6*** y reafirma la circunferencia verde haciendo una conversión completa al espacio Sparse.

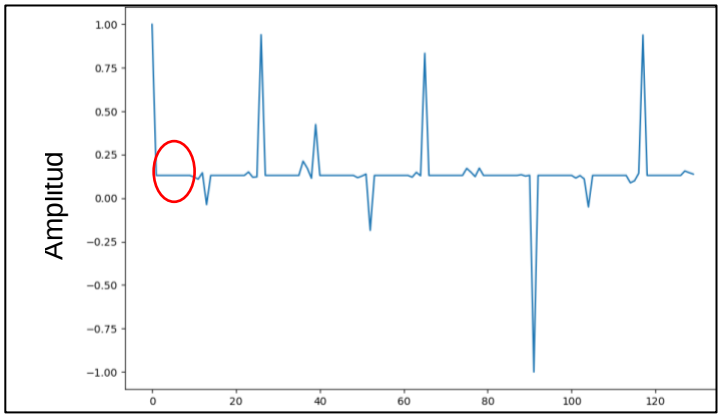


Figura 3.5 Señal en Espacio Sparse con Nivel de Offset

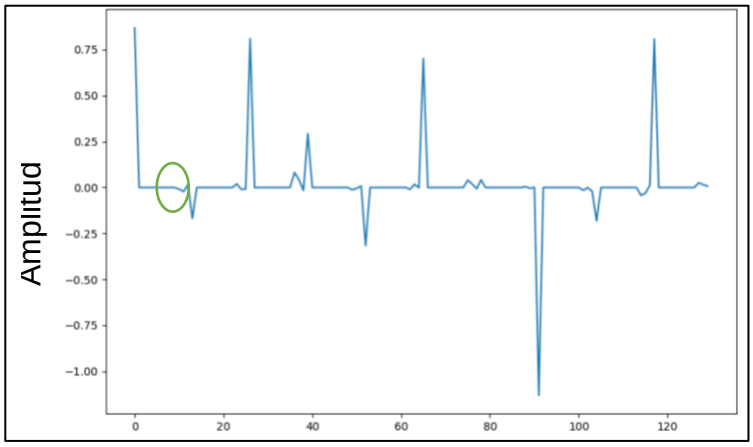


Figura 3.6 Señal en Espacio Sparse sin nivel de Offset

3.1.3 Compressive Sensing

La **Figura *3.7***, enseña los patrones extraídos para una secuencia de insolación disminuyendo su longitud en un 60% usando como matriz de compresión una matriz identidad a la cual se le aplicó transformada inversa discreta de coseno, mientras que la **Figura *3.8*** tanto como **Figura *3.9*** indica una reducción de dimensiones de un 70% y 80% respectivamente.

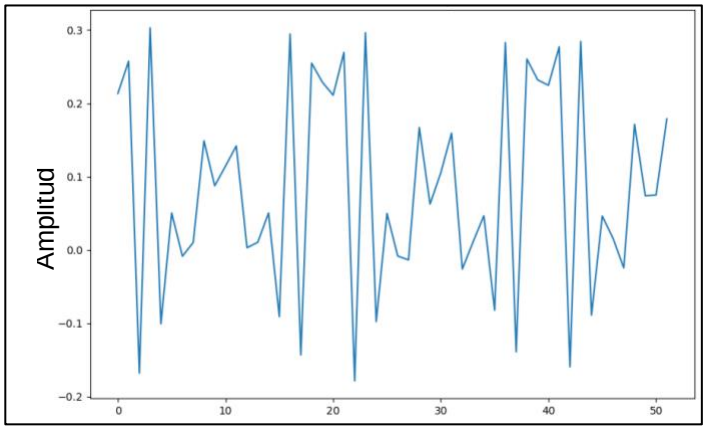


Figura 3.7 Extracción de Patrones con 60% de compresión usando Matriz Identidad junto a Transformada Coseno Inversa

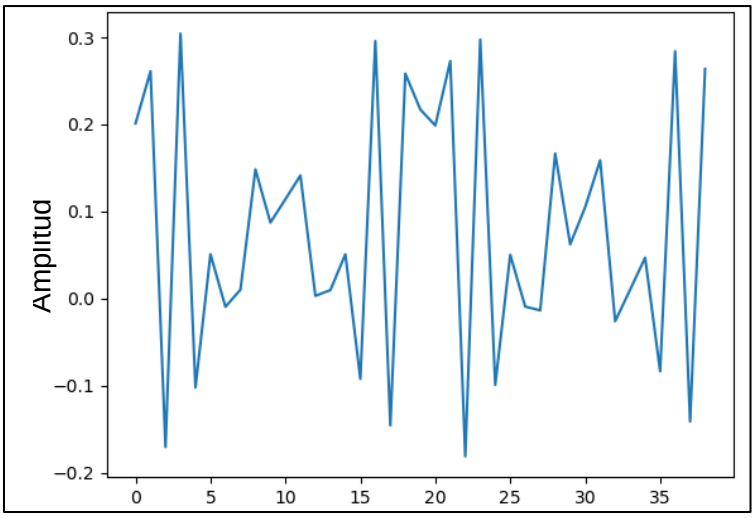


Figura 3.8 Extracción de Patrones con 70% de compresión usando Matriz Identidad junto a Transformada Coseno Inversa

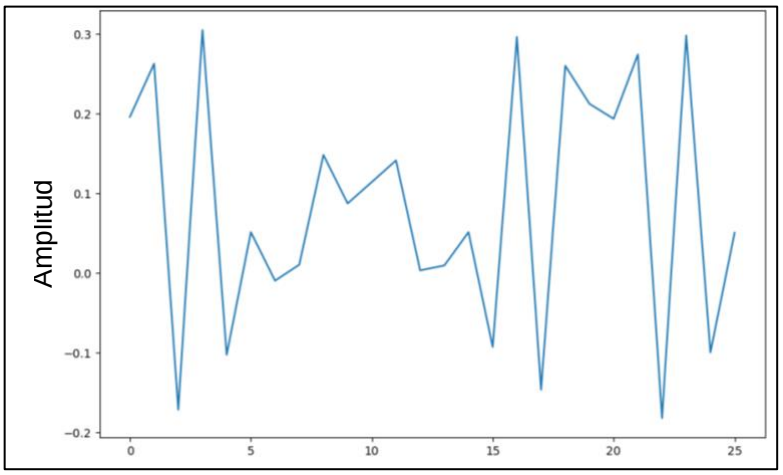


Figura 3.9 Extracción de Patrones con 80% de compresión usando Matriz Identidad junto a Transformada Coseno Inversa

La **Figura *3.10*** indica la obtención de características al operar por una matriz identidad lo que genera el mismo vector sparse (ya que una matriz identidad es el elemento unitario para los arreglos de 2 dimensiones) pero cortado en un 40%, 30% y 20%, correspondientes a cada nivel de compresión

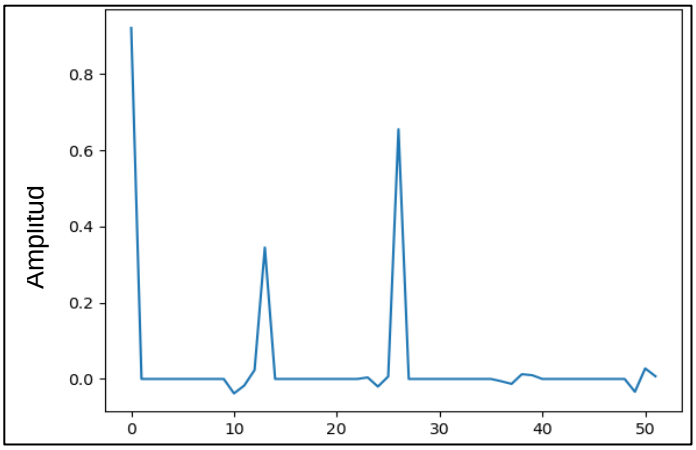


Figura 3.10 Extracción de Patrones con 60% de compresión usando Matriz Identidad

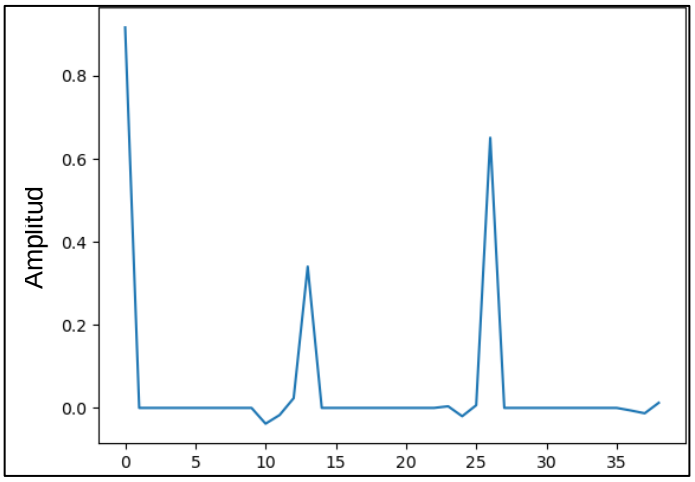


Figura 3.11 Extracción de Patrones con 70% de compresión usando Matriz Identidad

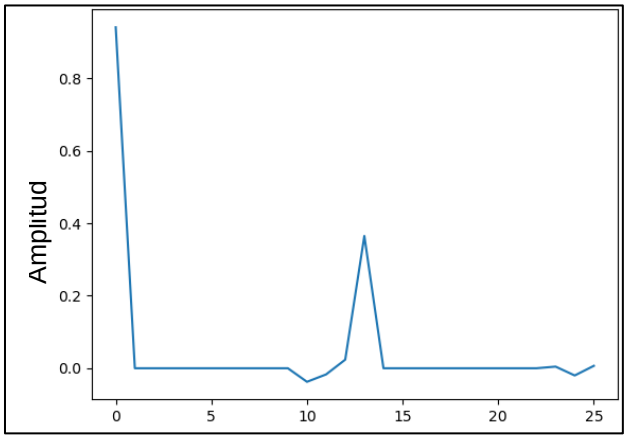


Figura 3.12 Extracción de Patrones con 80% de compresión usando Matriz Identidad

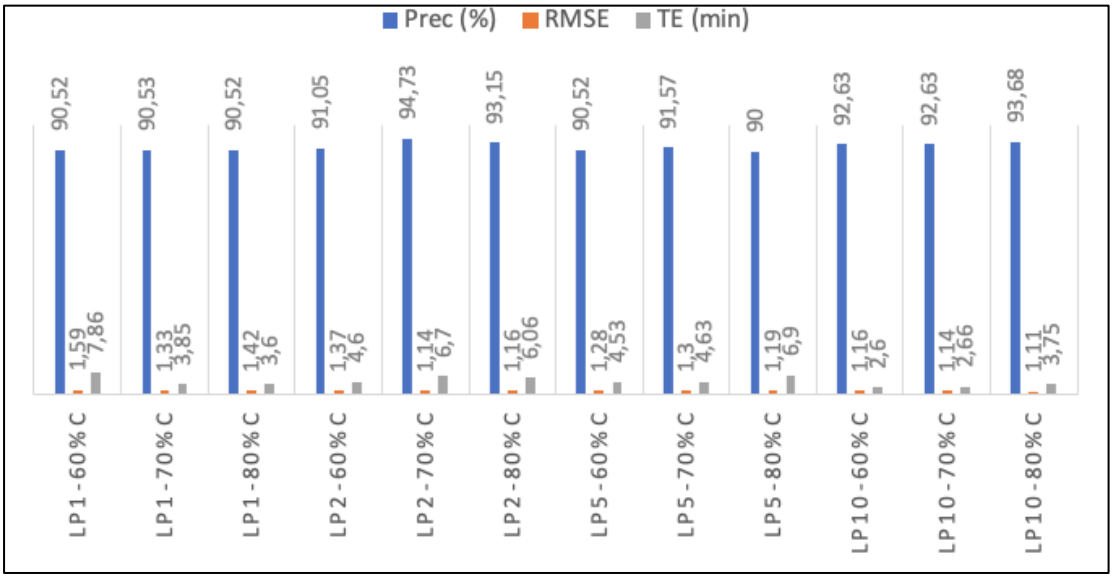
3.1.4 Entrenamiento y predicción

A continuación, se enseña un conjunto de tabla con los resultados obtenidos la cuales tienen unas siglas con los siguientes significados

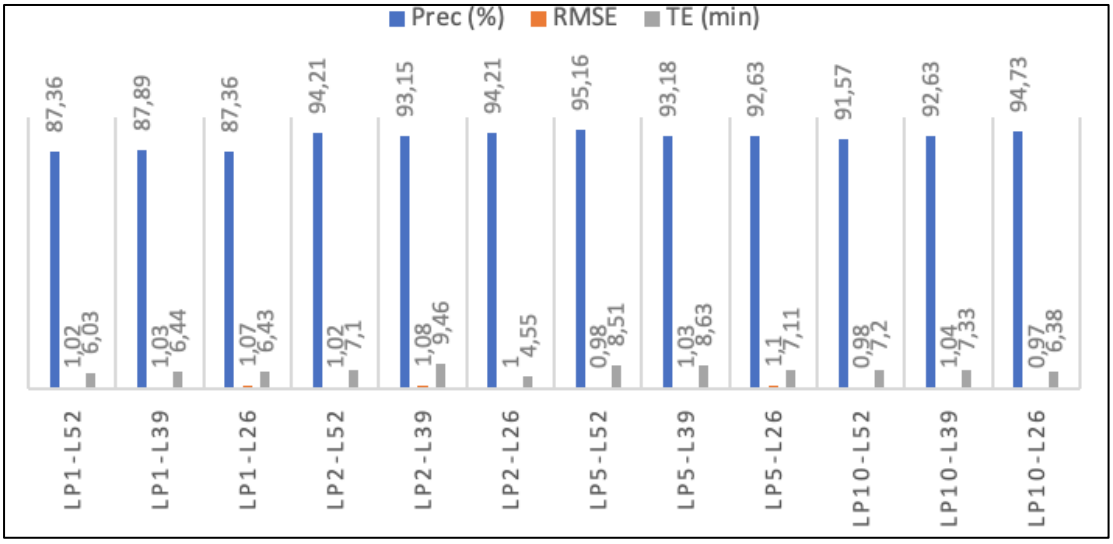
Lp-n = Longitud de Predicción n, donde n = 1,2,5,10
Comp-CS = compresión de Compressive Sensing
Prec = precisión
TE = Tiempo de Entrenamiento
L.-n = Secuencia de longitud n, donde n = 26, 39, 52
n%C = n% de compresión, donde n= 60, 70, 80

El conjunto de resultados de la **Figura *3.13*** hasta la **Figura *3.17*** enmarcan las estadísticas de la arquitectura 1, la parte (a) ilustra los números al usar CS mientras que la parte (b) de cada figura determina cuando se descartó esta técnica, cada sección cuenta con 3 columnas barras correspondientes a la porcentaje de precisión, RMSE y Tiempo de entrenamiento para cada longitud de predicción y nivel de compactación , para este caso la matriz de compresión corresponde a la transformada inversa del coseno de un arreglo diagonal, la patrones se extraen casi a la misma velocidad e incluso a un mayor que cuando no se implementó dicha extracción de patrones además los porcentajes de precisión se mantuvieron en un rango de valores cercanos.

La **Figura *3.13*** enseña los resultados de la Temperatura con la arquitectura 1 donde al aplicar CS parte (a) hubo mejor predicción en menor costo computacional en relación a cuando no se usó parte (b) donde el tiempo de entrenamiento **TE** tiene valores por encima.



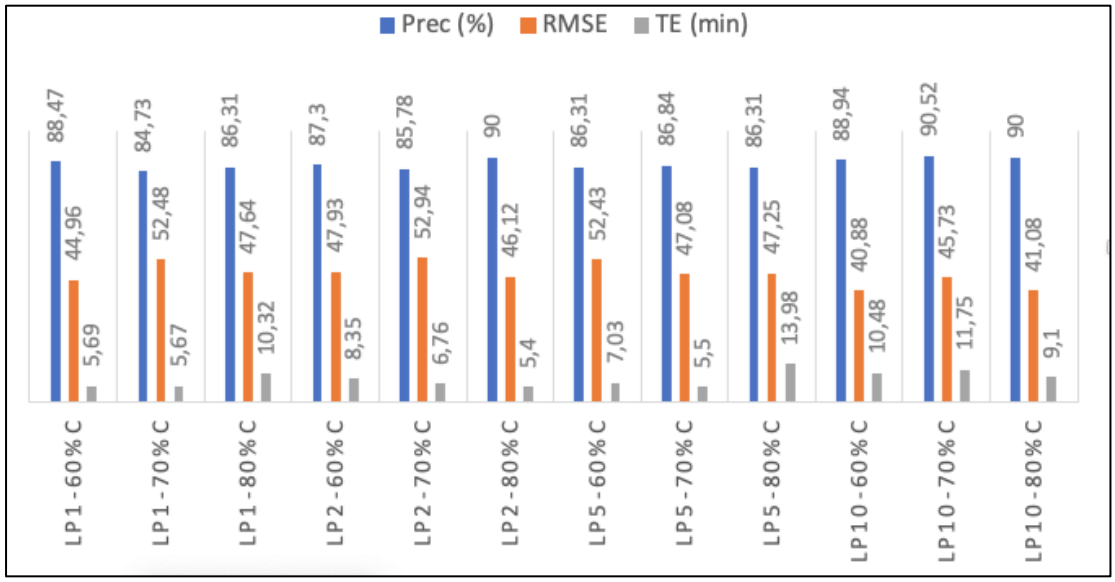
(a) Gráfico Estadísticas de predicción Temperatura con CS



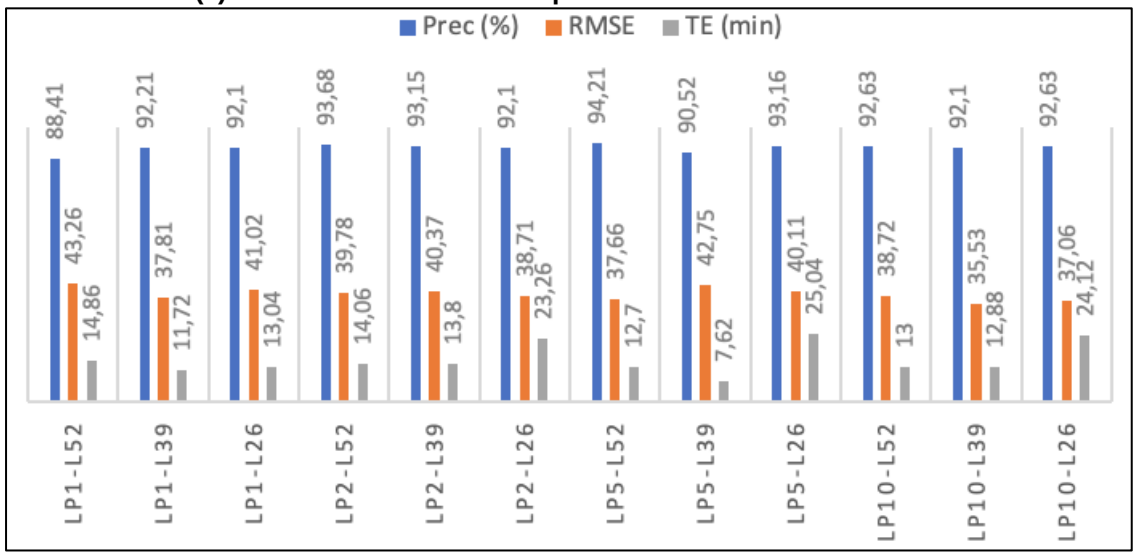
(b) Gráfico Estadísticas de predicción Temperatura sin CS

Figura 3.13 Estadísticas de Predicción del Temperatura ±2ºC Arquitectura 1- Ubicación 1

La **Figura *3.14*** enseña las estadísticas de la Humedad relativa donde al usar CS parte (b) hay una menor precisión con respecto a su omisión parte (a) aunque esta última necesitó mayor costo computacional, en general el RMSE aumentar por la magnitud y variabilidad de lo datos que oscila entre 90 y 60 unidades.



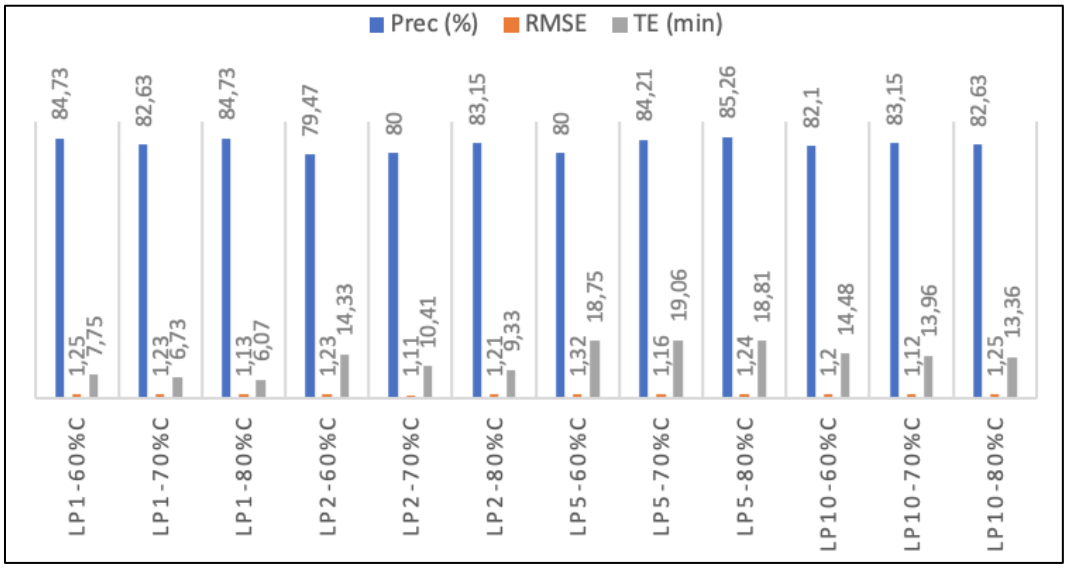
(a) Gráfico Estadísticas de predicción Humedad Relativa con CS



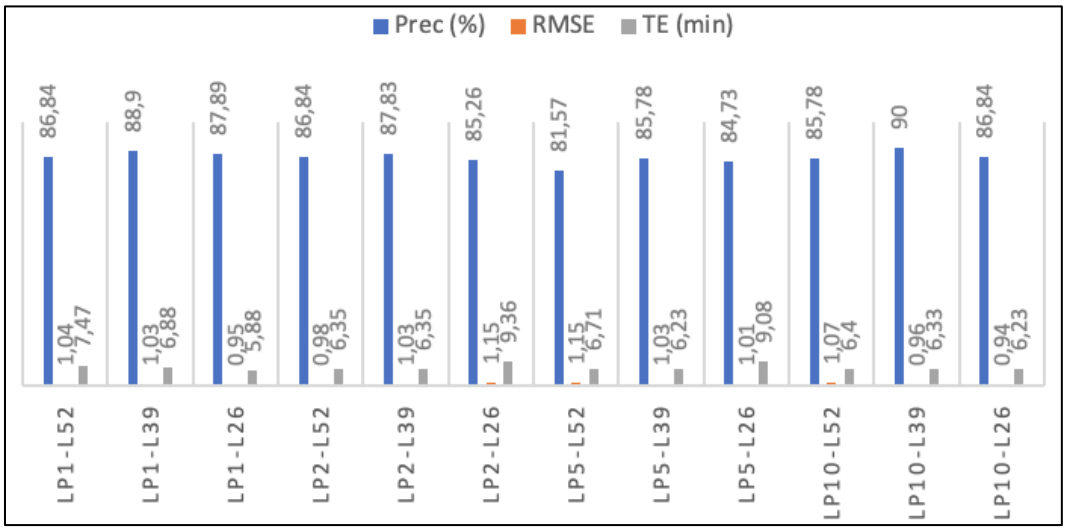
(b) Gráfico Estadísticas de predicción Humedad Relativa sin CS

Figura 3.14 Estadísticas de Predicción de Humedad Relativa ±10% Arquitectura 1-Ubicación 1

La ***Figura 3.15*** ilustra porcentajes de precisión similares para la velocidad del viento para ambas condiciones, pero contradictoriamente al aplicar CS el tiempo de procesamiento **TE** y el error cuadrático medio **RMSE** fueron mayores



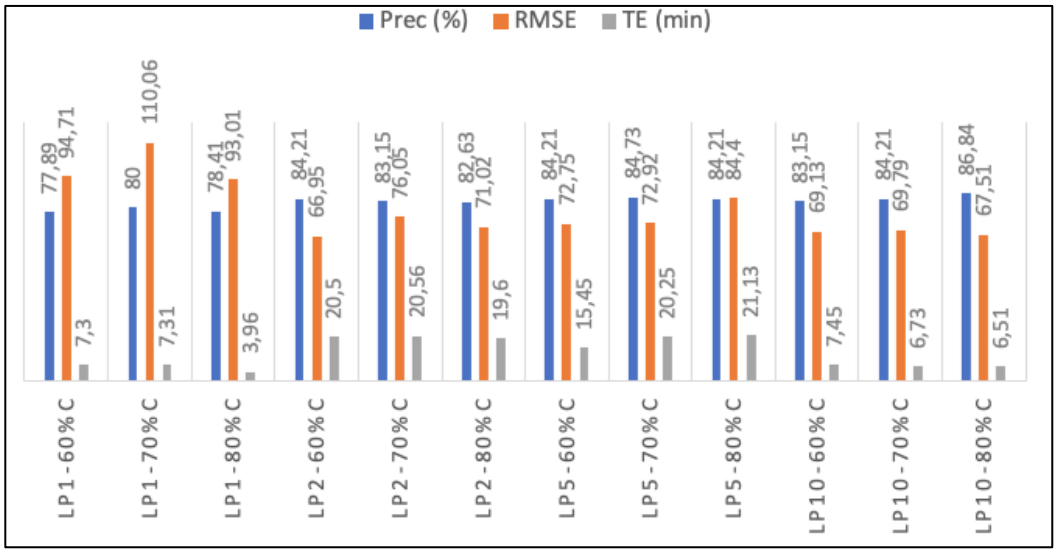
(a) Gráfico Estadísticas de predicción Velocidad del Viento con CS



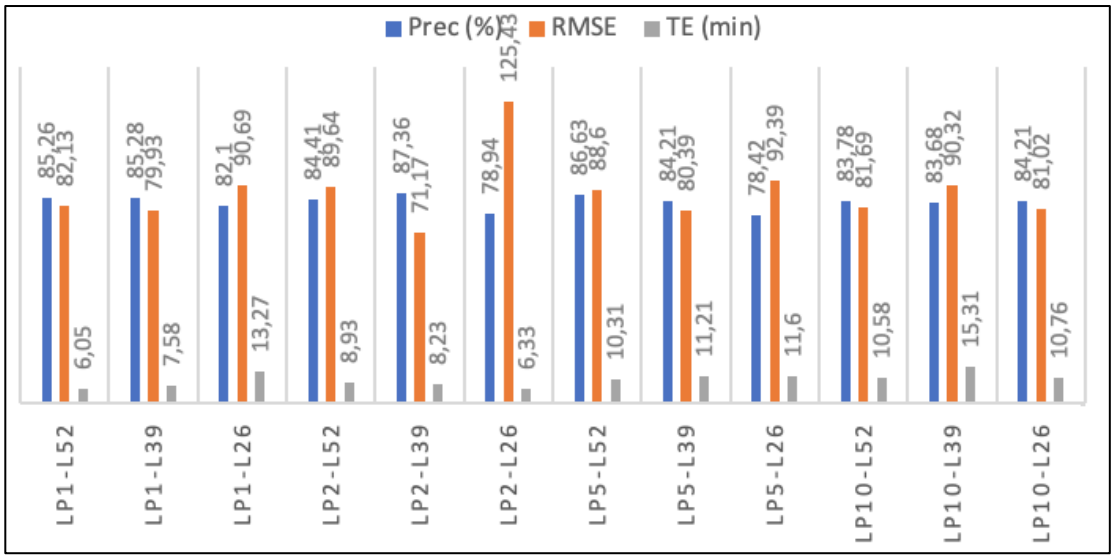
(b) Gráfico Estadísticas de predicción Humedad Relativa sin CS

Figura 3.15 Estadísticas de Predicción de Velocidad del Viento ±1,5m/s Arquitectura 1-Ubicación 1

La precipitación presentó los mayores valores de RMSE teniendo magnitudes mayores a 100 para casos de (a) y (b), la efectividad de predicciones fue discreta como se ve en la **Figura *3.16*** ,donde ningún porcentaje llega al 90 %.



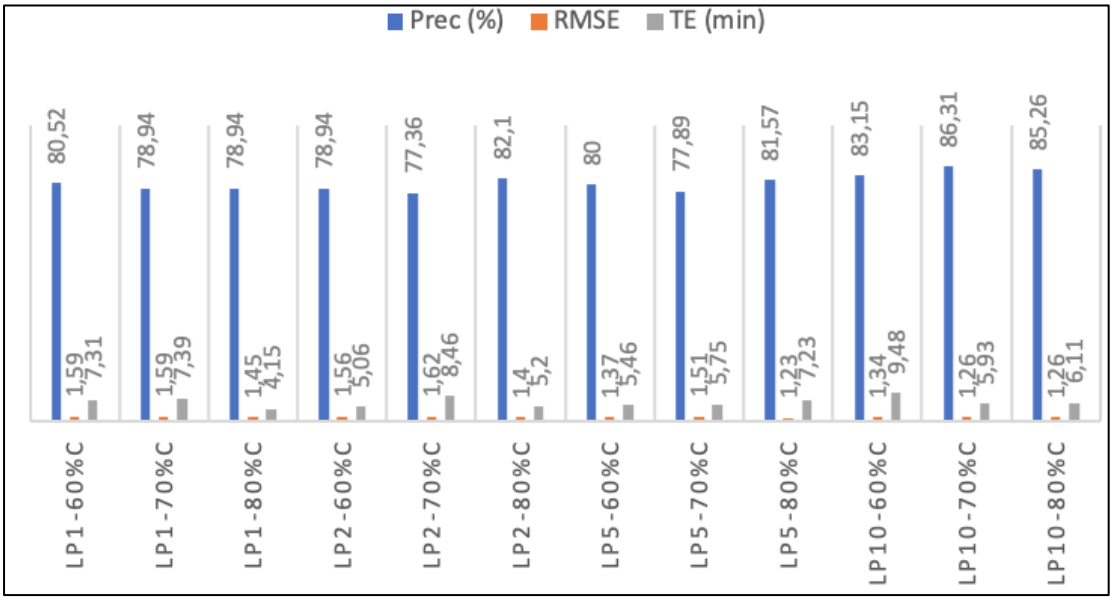
(a) Gráfico Estadísticas de predicción Precipitación con CS



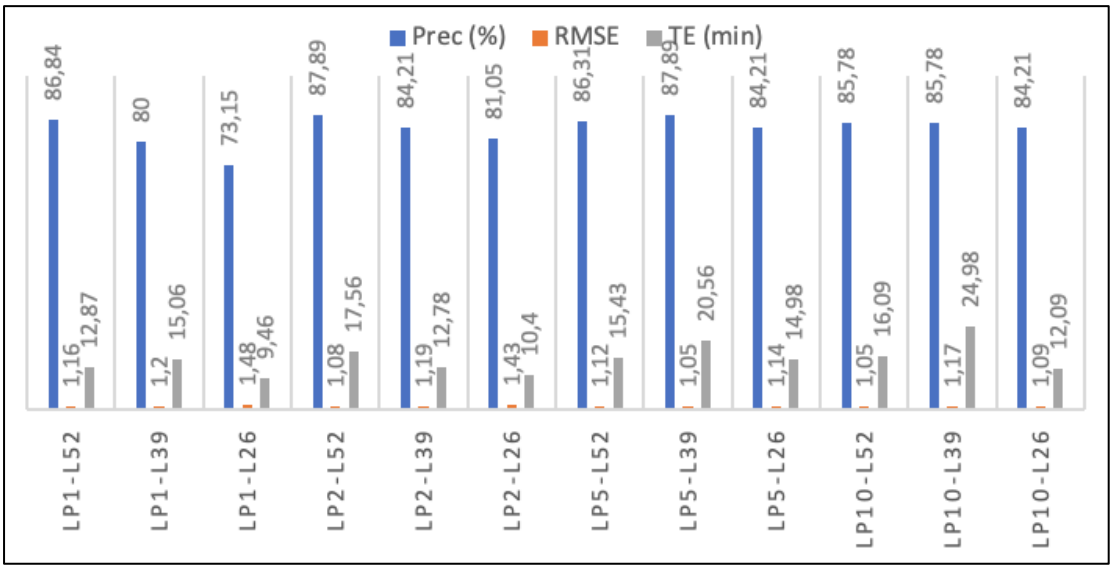
(a) Gráfico Estadísticas de predicción Precipitación sin CS

Figura 3.16 Estadísticas de Predicción de Precipitación ±10mm/día Arquitectura 1-Ubicación 1

Según la **Figura *3.17*** para la insolación el descartar CS parte(b) presentó un rendimiento un poco mejor esto se ve reflejado en la similitud del RMSE para todas las pruebas; aunque parte (a) tiene un menor rendimiento computacional.



(a) Gráfico Estadísticas de predicción insolación con CS

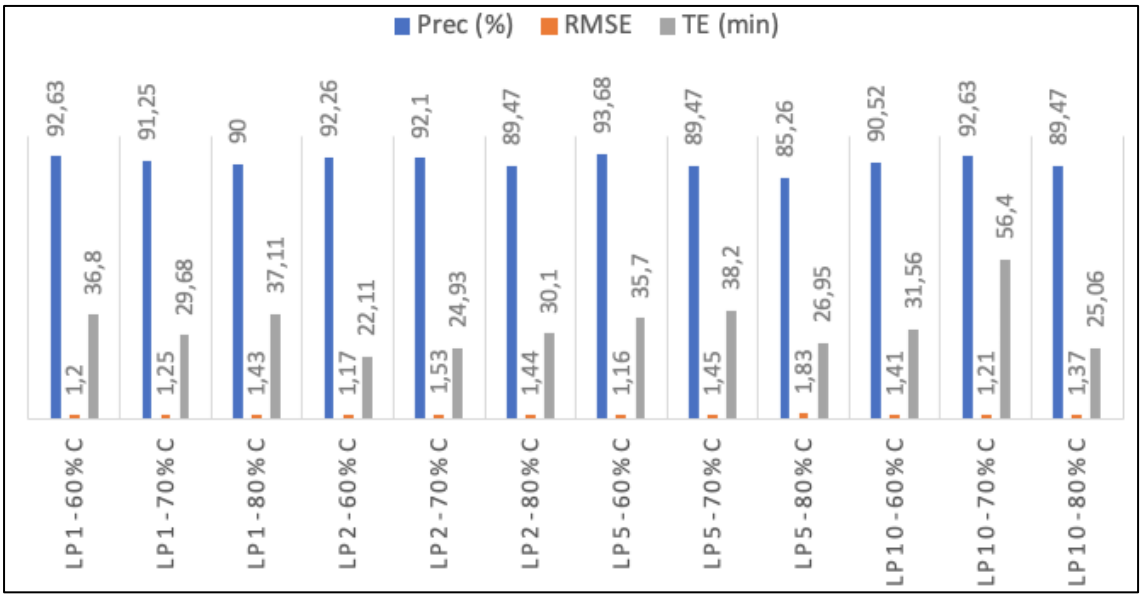


(b) Gráfico Estadísticas de predicción insolación sin CS

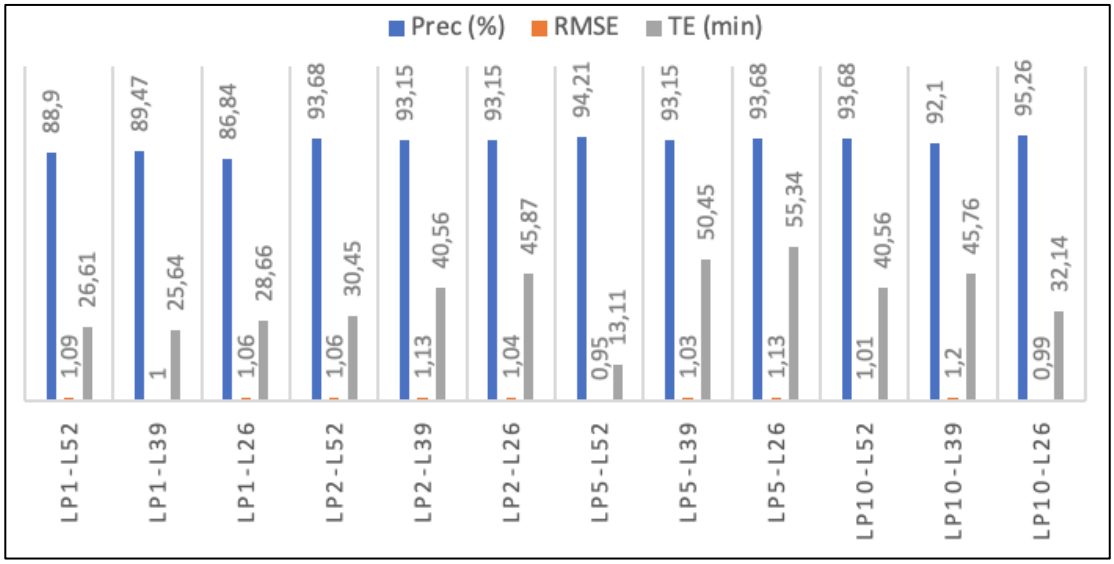
Figura 3.17 Estadísticas de Predicción de Insolación ±1.5kwh/m2 Arquitectura 1-Ubicación 1

Mientras que desde la **Figura *3.18*** hasta la **Figura *3.22*Error! Reference source not found.** se presenta el segundo modelo donde se nota una permanencia en los porcentajes de precisión y valores de **RMSE** pero con un aumento en el tiempo de entrenamiento al usar CS, con respecto a cuándo se descartó la compresión, esta modelo extrae los patrones de una forma un poco más lenta.

Como se ve en la **Figura *3.18*** la arquitectura 2 presentó mayor tiempo de entrenamiento y precisiones similares lo que va de la mano al RMSE, en comparación primera estructuración de la red neuronal.



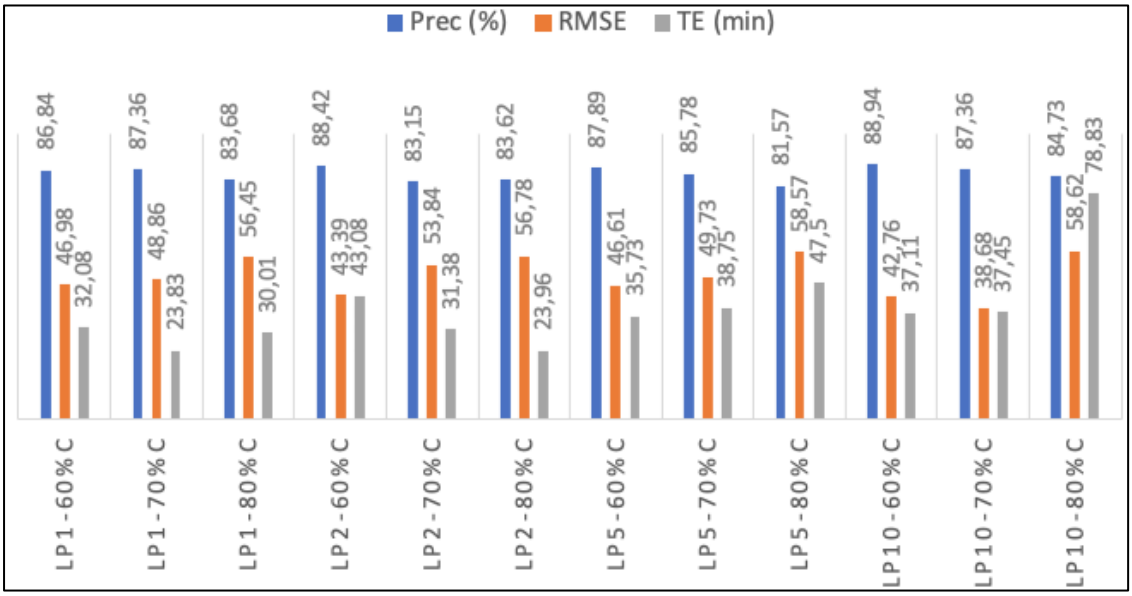
(a) Gráfico Estadísticas de predicción Temperatura con CS



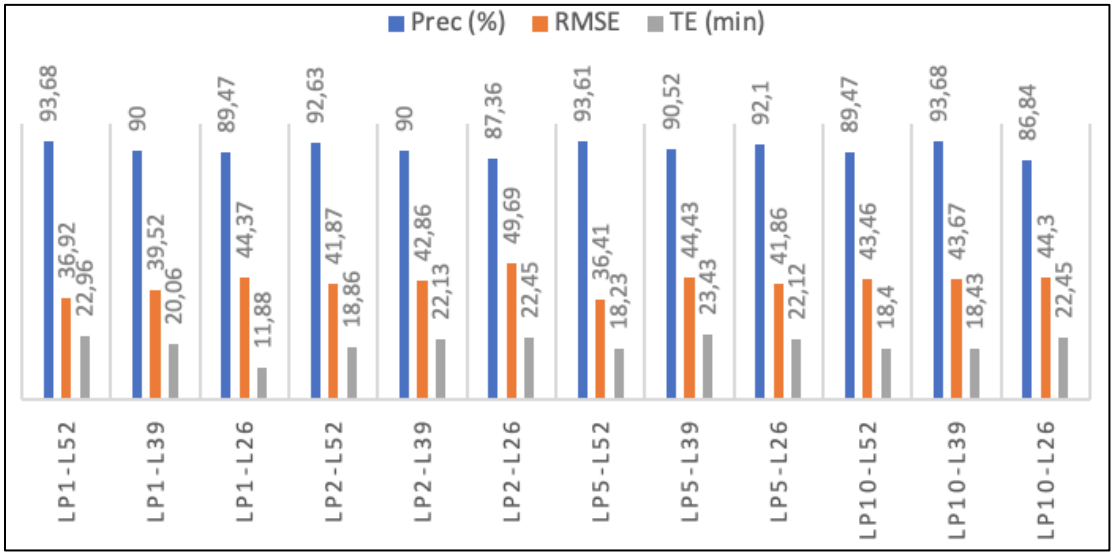
(b) Gráfico Estadísticas de predicción Temperatura sin CS

Figura 3.18 Estadísticas de Temperatura ±2ºC Arquitectura 2- Ubicación 1

Como enseña la **Figura *3.19*** para esta arquitectura no usar CS presentó menor costo para una mayor precisión. Los valores se RMSE se conservan con relación a la arquitectura 1.



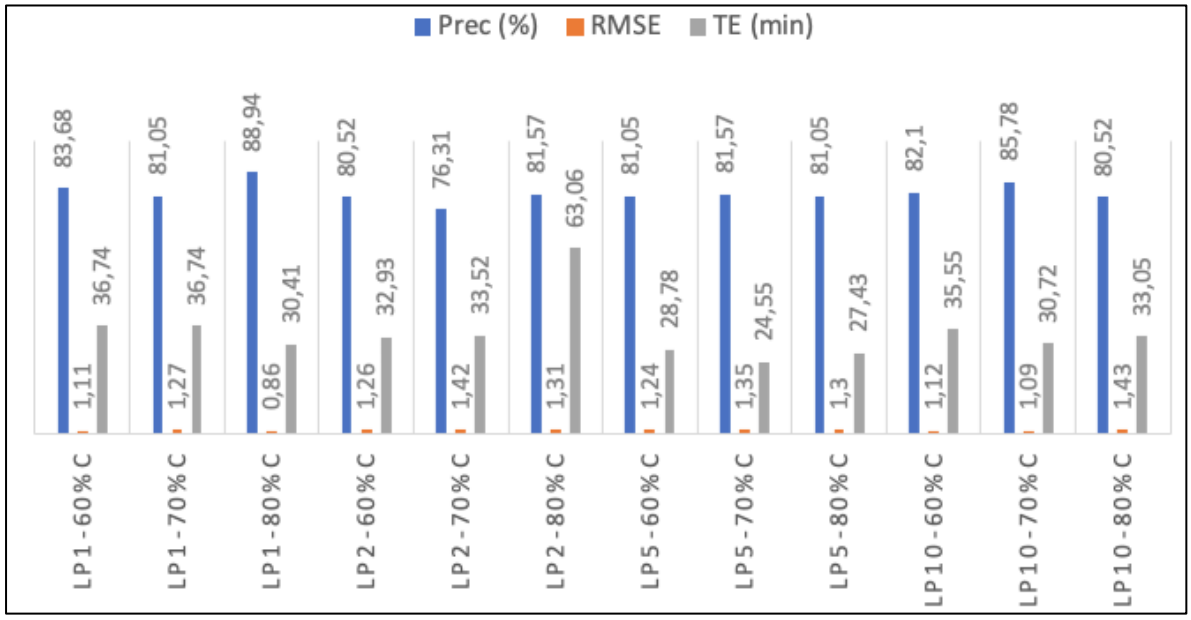
(a) Gráfico Estadísticas de predicción Humedad Relativa con CS



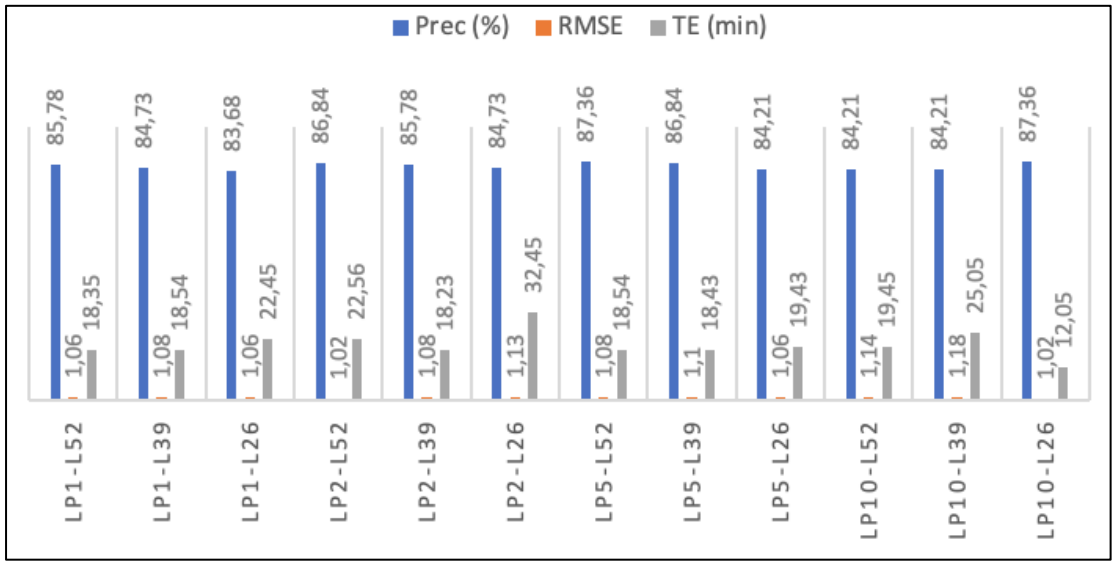
(b) Gráfico Estadísticas de predicción Humedad Relativa sin CS

Figura 3.19 Estadísticas de Humedad Relativa ±10% Arquitectura 2- Ubicación 1

De acuerdo a la **Figura *3.20*** se conserva las características de efectividad y RMSE con la tardanza de la arquitectura 2.



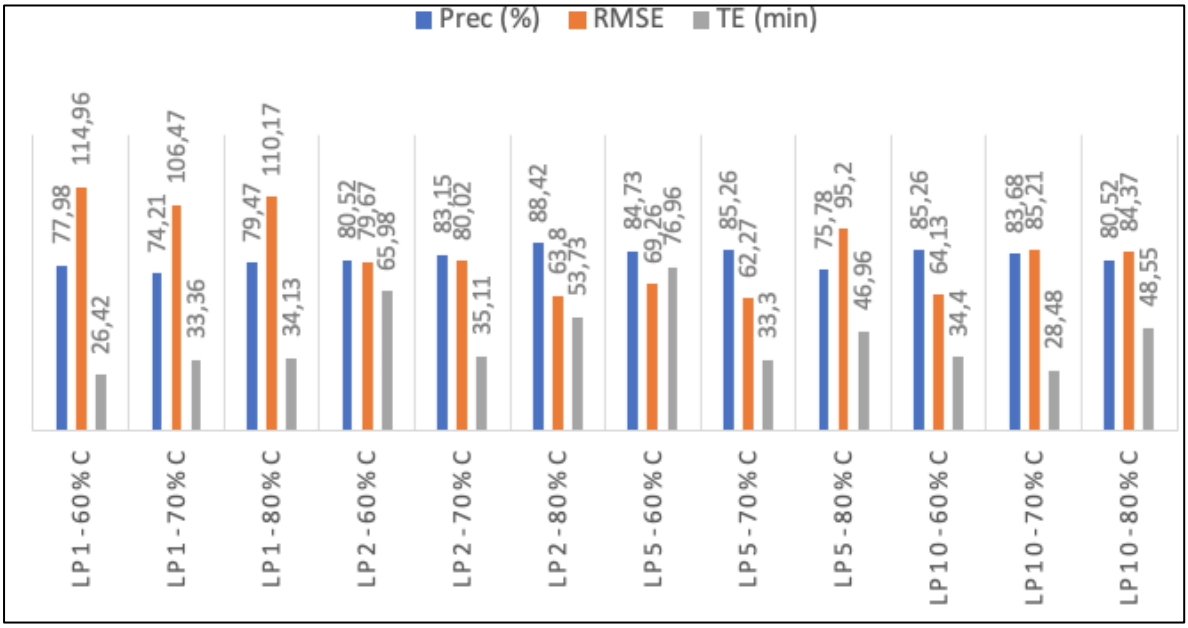
(a) Gráfico Estadísticas de predicción Velocidad del Viento con CS



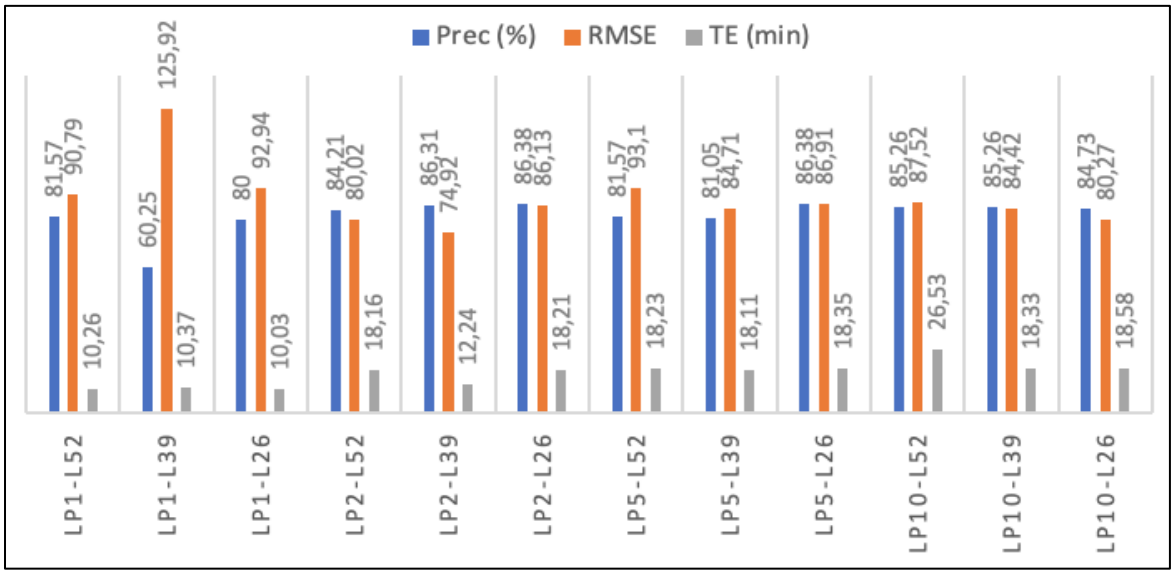
(b) Gráfico Estadísticas de predicción Velocidad del Viento sin CS

Figura 3.20 Estadísticas de Velocidad del Viento ±1.5m/s Arquitectura 2- Ubicación 1

Para la **Figura *3.21*** los valores mayores de RMSE se presentaron para una longitud de predicción simple y a nivel general la precisión no es muy elevada, el entrenamiento con CS implicó un mayor tiempo.



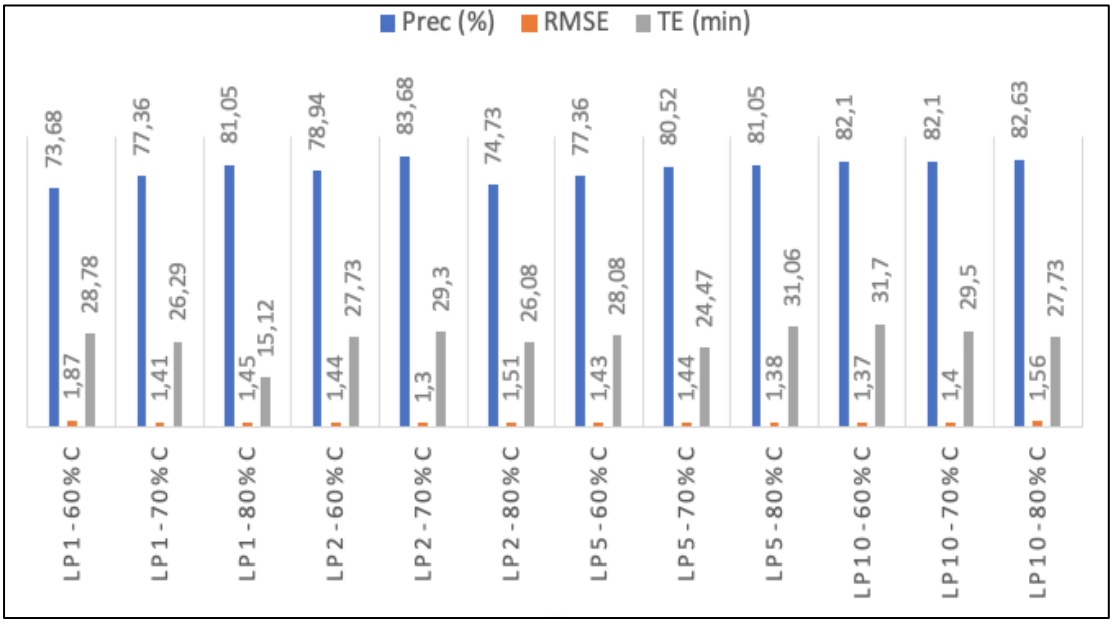
(a) Gráfico Estadísticas de predicción Precipitación con CS



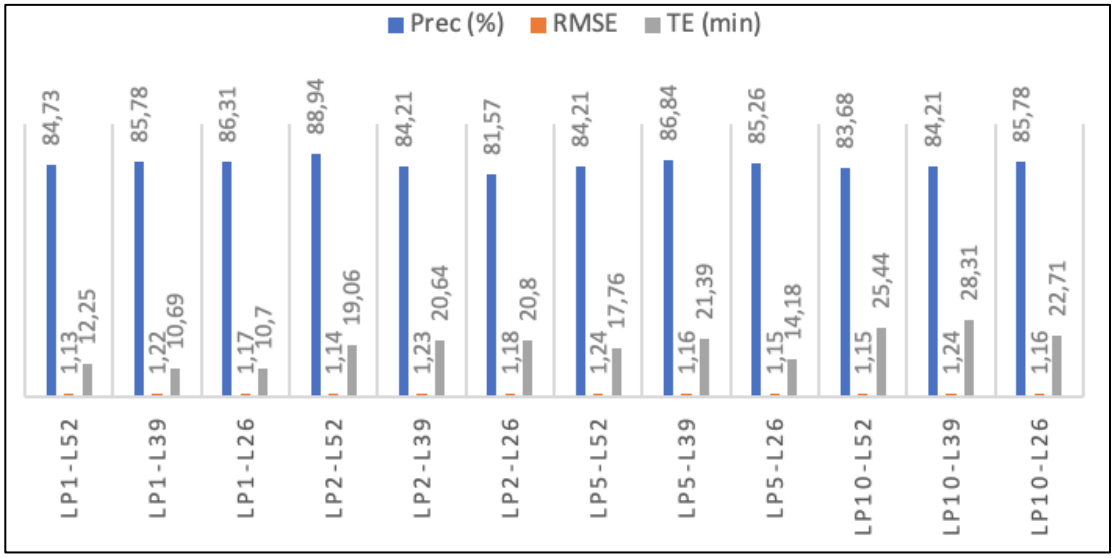
(b) Gráfico Estadísticas de predicción Insolación con CS

Figura 3.21 Estadísticas de Precipitación ±10mm/día Arquitectura 2- Ubicación 1

Para la **Figura *3.22*** la precisión se vio favorecida para cuando se ingresaron las secuencias de tiempo originales para el entrenamiento del modelo, pero los aciertos con CS no estuvieron muy por debajo, aunque los tiempos de aprendizaje si se elevaron en gran medida.

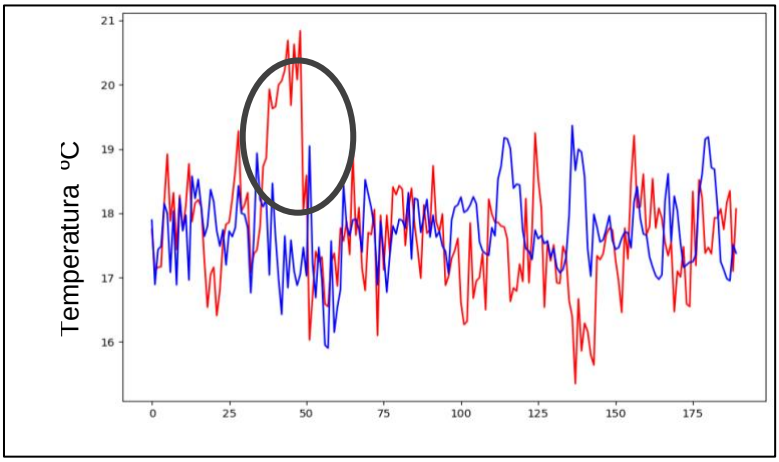


(a) Gráfico Estadísticas de predicción Insolación con CS

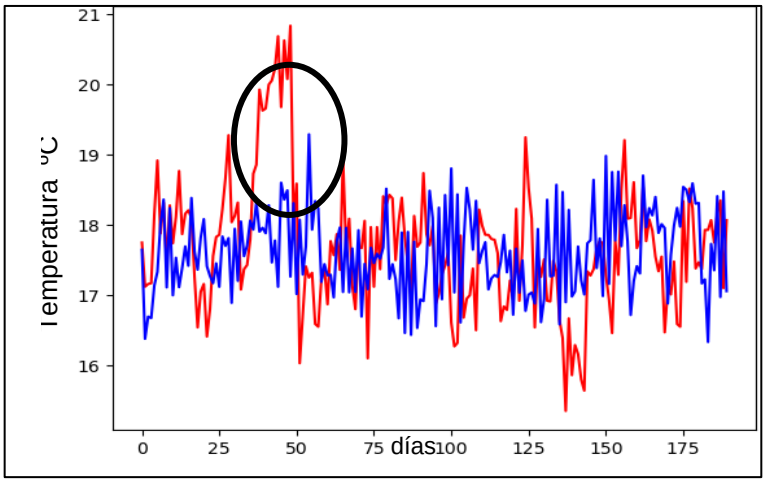


(b) Gráfico Estadísticas de predicción Insolación sin CS

Figura 3.22 Estadísticas de Insolación ±1.5kwh/m2 Arquitectura 2- Ubicación 1



(a) Predicción de Temperatura Simple a un 80% de compresión CS modelo 1 (Rojo=Real – Azul=Estimada)[Elaboración Propia]



(b) Predicción de Temperatura Múltiple (2 valores) a un 80% de compresión CS modelo 1 (Rojo=Real – Azul=Estimada) [Elaboración Propia]

Figura 3.23 Predicción de Temperatura Simple y Múltiple

La **Figura *3.23*** ilustra la diferencia sustancial entre una predicción simple con una múltiple, se puede observar que en la parte (a) las oscilaciones son menores y la línea de tendencia es más limpia, en cambio, en la parte (b) se predicen 10 datos de inmediato lo que crea más fluctuaciones en la respuesta además algo que tiene en común ambas a nivel general y que incluso se presenta en la humedad relativa, es la incapacidad de predecir los picos o valles abruptos y duraderos como encierran en círculos negros en ambas representaciones.

La **Figura *3.24*** ilustra una predicción sin aplicar la compresión donde se puede notar que una característica al descartar CS es la incapacidad de detectar cambios tanto sencillos como abruptos, se observa que la gráfica sigue un línea de tendencia cercano a la media de los datos.

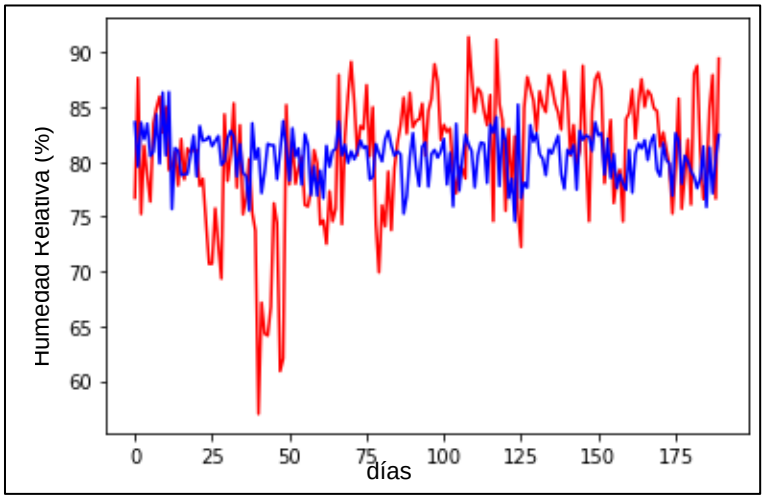


Figura 3.24 Predicción de la Humedad Relativa simple sin CS Longitud de Secuencia de 26 [Elaboración Propia] *(Rojo=Real – Azul=Estimada)*

La precipitación es una de las variables más complejas de predecir que usualmente se apoya en parámetros con fuerza de relación en este se tomaron como series de tiempo con una respuesta discreta fallando en algunos puntos con alta probabilidad de lluvia como se ve en la **Figura *3.25*** y marcando otros falsos positivos.

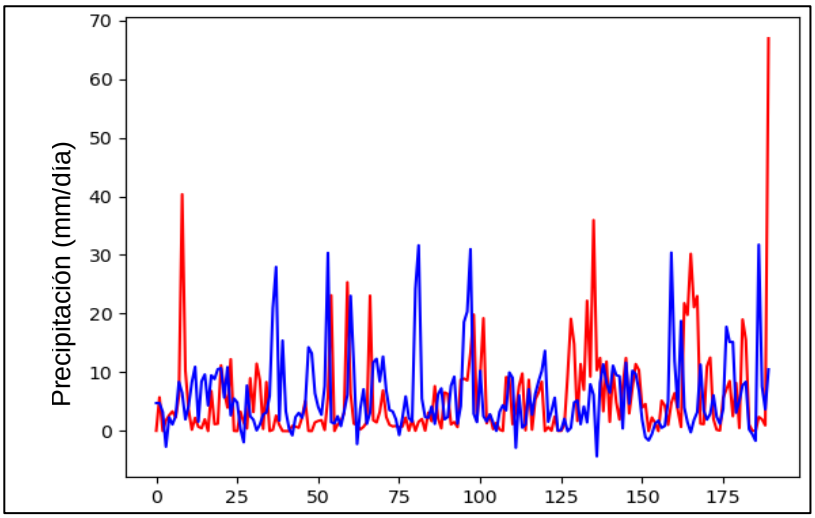


Figura 3.25 Predicción de Precipitación Total Múltiple con Modelo 1 Compresión del 70% *(Rojo=Real – Azul=Estimada)*

Se realizaron pruebas de igual forma en una ubicación diferente correspondientes a las coordenadas de latitud = 16.3557 y longitud = -80.2263 donde se evaluó el modelo planteado desde el entrenamiento, ya que cada lugar posee unas características meteorológicas distintas, las estadísticas se evaluaron para una predicción simple y una múltiple con longitud de etiquetas de 2 tal como se desde la **Figura *3.26*** hasta la **Figura *3.30***

Para la ubicación 2 la **Figura *3.26*** y **Figura *3.27***, ilustran los más alto indices de rendimiento y precisión, para humedad relativa y temperatura, además la toleracia se disminuyó para este conjunto datos. Por la efectividadlos dos casos.

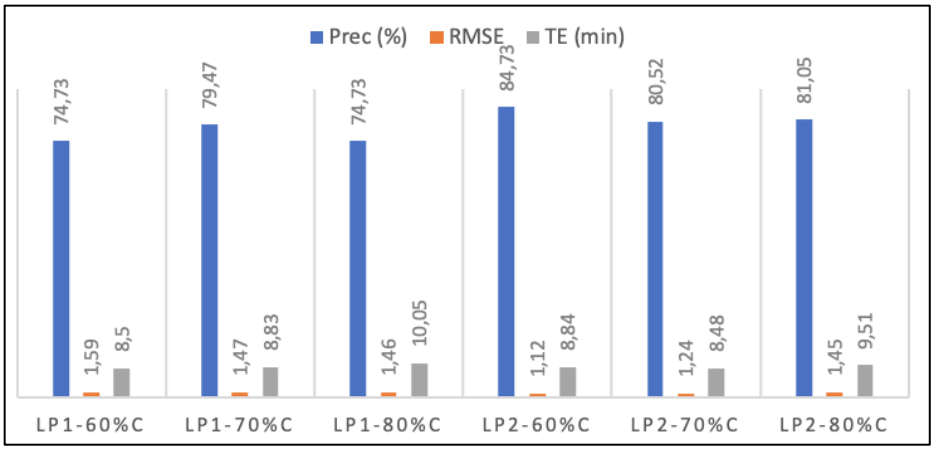


Figura 3.26 Estadísticas Predicción Temperatura ±1.5 ºC-Arquitectura 1-Ubicación 2

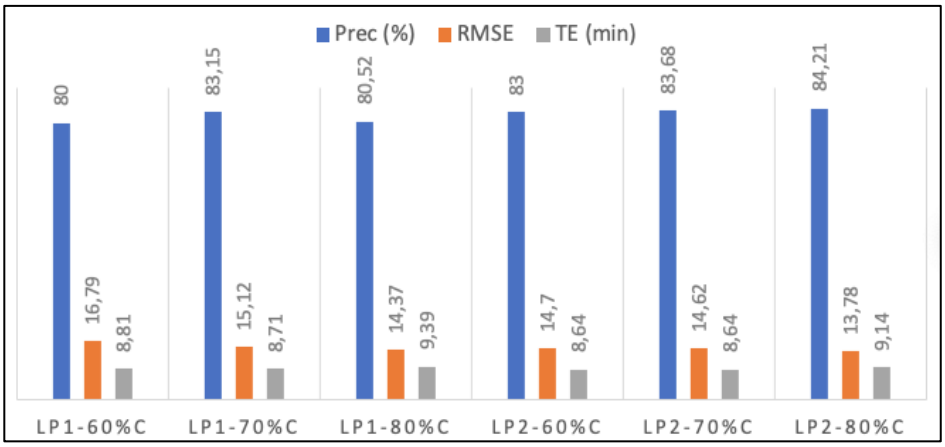


Figura 3.27 Estadísticas Predicción Humedad Relativa ±5 %-Arquitectura 1-Ubicación 2

Mientras que variables como la insolación, precipitación y velocidad del viento descritas en la **Figura *3.28***, **Figura *3.29*** y **Figura *3.30*** tiene bajos porcentajes de valores acertados, para la precipitación se redujo la toleración pero para insolación y velocidad del viento se aumento en 0.5 por la complejidad del comportamiento, aunque se conservó el bajo tiempo de procesamiento.

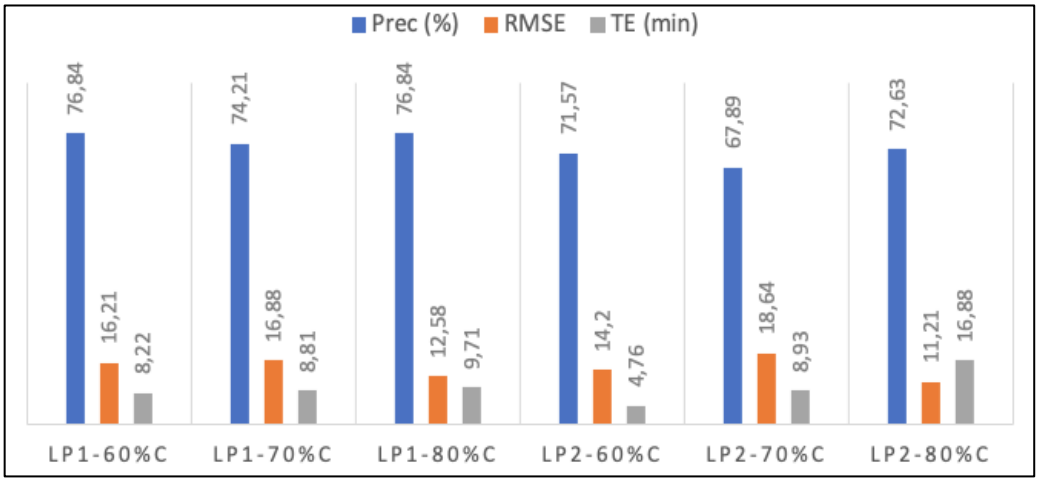


Figura 3.28 Estadísticas Predicción Precipitación ±2mm/día-Arquitectura 1-Ubicación 2

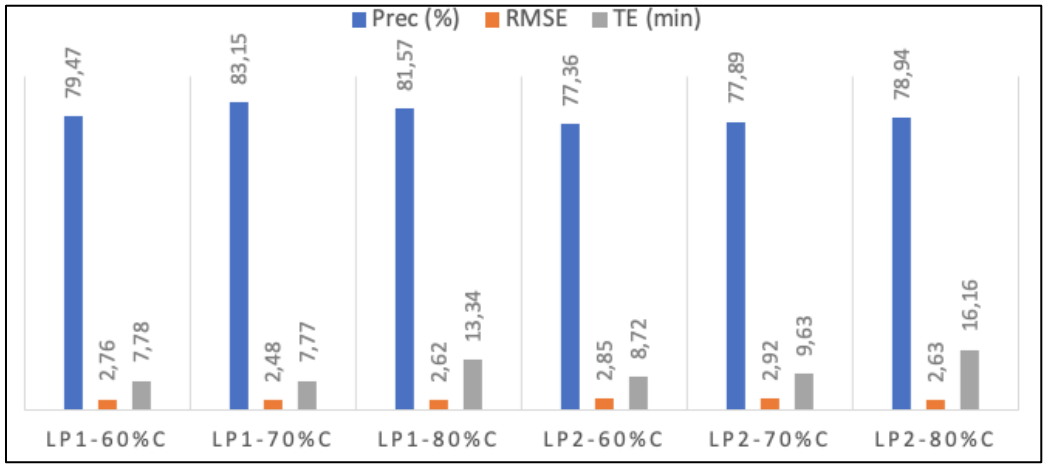


Figura 3.29 Estadísticas Predicción Insolación ±2kwh/m2-Arquitectura 1-Ubicación 2

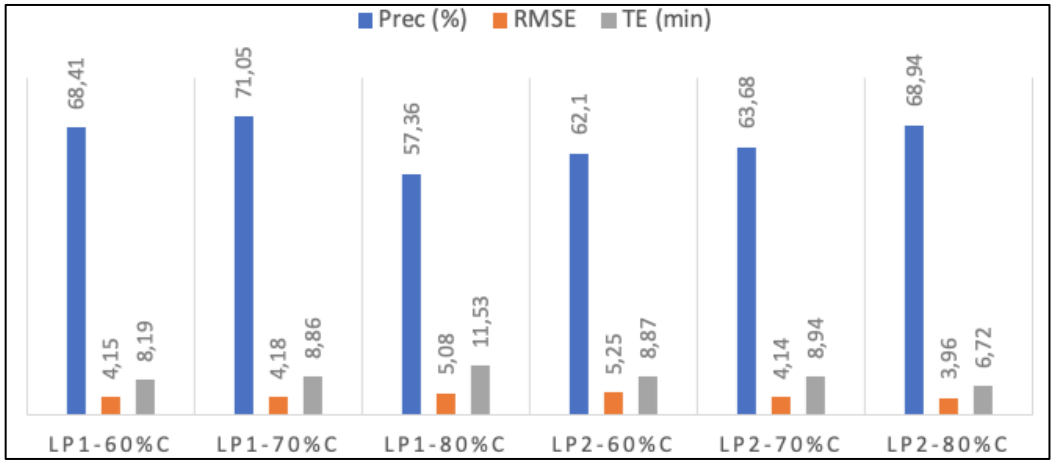


Figura 3.30 Estadísticas Predicción Velocidad del Viento ±2ms-Arquitectura 1-Ubicación

Tabla 3.1 Predicción aleatorias mes de mayo ubicación 1 Temperatura

Fecha	Real (º C)	Pronosticada (ºC)	Error
01-mayo-2020	18.5	17.96	2.91 %
14-mayo-2020	18.01	17.29	3.9%
30-mayo-2020	17.21	16.66	3.19%

Tabla 3.2 Predicción aleatorias mes de mayo ubicación 1 Velocidad del Viento

Fecha	Real (m/s)	Pronosticada (m/s)	Error
01-mayo-2020	1.68	0.6	64.28%
14-mayo-2020	2.01	2.42	20.39%
30-mayo-2020	1.14	1.61	41.22%

Tabla 3.3 Predicción aleatorias mes de mayo ubicación 1 Humedad Relativa

Fecha	Real (%)	Pronosticada (%)	Error
01-mayo-2020	89.51	79.82	10.82%
14-mayo-2020	87.44	84.62	3.2%
30-mayo-2020	85.96	81.56	3.2%

Tabla 3.4 Predicción aleatorias mes de mayo ubicación 1 Precipitación

Fecha	Real (mm/día)	Pronosticada (mm/día)	Error
01-mayo-2020	15.27	6.51	57.36%
14-mayo-2020	9.1	8.6	5.4%
30-mayo-2020	13	2.13	82.30%

Tabla 3.5 Predicción aleatorias mes de mayo ubicación 1 Insolación

Fecha	Real (Kwh/m2)	Pronosticada (Kwh/m2))	Error
01-mayo-2020	3.46	5.43	56.93%
14-mayo-2020	4.46	4.58	2.69%
30-mayo-2020	3.74	5.09	36.10%

Desde la ***Tabla 3.1*** a la ***Tabla 3.5*** se muestra mediciones puntuales para 3 determinadas y la diferencia entre la predicción junto al valor real, observando que para puntos específicos se pueden calcular datos erróneos pero que en la sumatoria de tiempo con una tolerancia adecuada se puede obtener una gráfica adecuada.

Tabla 3.6 Predicción aleatorias mes de mayo ubicación 2 Temperatura

Fecha	Real (º C)	Pronosticada (ºC)	Error
01-mayo-2020	28.09	28.38	2.91 %
14-mayo-2020	28.12	27.4	3.9%
30-mayo-2020	28.37	27.85	3.19%

Tabla 3.7 Predicción aleatorias mes de mayo ubicación 2 Humedad Relativa

Fecha	Real (%)	Pronosticada (%)	Error
01-mayo-2020	74.03	82.44	11.36%
14-mayo-2020	79.23	89.35	12.6%
30-mayo-2020	83.16	77.97	6.24%

Tabla 3.8 Predicción aleatorias mes de mayo ubicación 2 Velocidad del Viento

Fecha	Real (m/s)	Pronosticada (m/s)	Error
01-mayo-2020	5.07	5.8	14.39%
14-mayo-2020	5.64	6.52	15.602%
30-mayo-2020	6.79	6.15	9.4%

Tabla 3.9 Predicción aleatorias mes de mayo ubicación 2 Precipitación

Fecha	Real (mm/día)	Pronosticada (mm/día)	Error
01-mayo-2020	0.16	1.16	625%
14-mayo-2020	0.89	6.68	650.3%
30-mayo-2020	2.23	1.49	33.18%

Tabla 3.10 Predicción aleatorias mes de mayo ubicación 2 Insolación

Fecha	Real (Kwh/m2)	Pronosticada (Kwh/m2))	Error
01-mayo-2020	7.42	5.5	25.8%
14-mayo-2020	7.22	3.94	45.42%
30-mayo-2020	7.09	5.88	21.72%

De la ***Tabla 3.6*** hasta ***Tabla 3.10*** se realizan las respuestas de fechas escogidas dentro del mes de mayo, donde en mediciones puntuales se ven los errores absolutos mas bajos para la temperatura.

Conclusiones

Se logró registrar las variables ambientales relevantes en la predicción del estado del tiempo tales como Temperatura, Humedad Relativa, Velocidad del Viento, Precipitación e Insolación, esta última se relacionó de manera inversa con la nubosidad para dar un veredicto de este parámetro la obtención automática se implementó mediante el uso de la Base de datos POWER NASA la cual suministra una API con su respectivo instructivo brindando la posibilidad de adquirir de forma de libre los datos meteorológicos a partir de 2015, la interfaz de programación de aplicaciones ayudó a implementar una mejor plataforma debido a que esta automatiza la adquisición de los datos mediante solicitudes web para posteriormente aplicar el modelo matemático planteado.

Se logró implementar una modelo de extracción de patrones usando Compressive Sensing teniendo como matriz de comprensión un arreglo bidimensional diagonal unitario y la transformada inversa discreta del coseno del mismo, acortando las secuencias de tiempo de longitud 130 hasta dimensiones de 26, 39 y 52 para compresiones del 80%, 70% y 60% conservando la información relevante de los datos destacando que la segunda matriz fue más efectiva en la extracción de patrones ya que un arreglo diagonal es proporcional a un producto por uno, por lo que resultaba el mismo vector Sparse, en este proceso no fue determinante el porcentaje de compactación del vector de entrenamiento en la obtención de patrones distintivos, repercutiendo únicamente en los tiempos de procesamiento, posteriormente el aprendizaje de dichas características se implementó usando una red neuronal multicapa con dos arquitecturas de 8 capas de los cuales se puede destacar que la función de activación ReLu le da mayor velocidad al entrenamiento mientras que Tangente Hiperbólica (tanh) fue apta para detectar cambios abruptos de la señal aunque a un ritmo más lento. Se puede afirmar que Compressive Sensing es una técnica apta para extraer patrones de series de tiempo complejas, dando resultados buenos para un historial de tiempo de tan solo 4 años y considerándolas dependientes de sí mismas. De igual forma se realizó la variante de etiquetas únicas y múltiples considerando estas últimas para dimensiones de 2, 5 y 10, sustancialmente la salida simple tenía como respuesta una línea de tendencia sin demasiadas oscilaciones, mientras que la compuesta cuenta con un número elevado de pequeñas variaciones, aunque manteniendo una predilección similar, a la hora de discriminar la forma de cada respuesta la simple es conveniente para el ejercicio ya que una señal con pocas oscilaciones permite notar mejor la orientación de cada parámetro. Se notó que al que al descartar CS los datos tendían a permanecer en la media del conjunto omitiendo los cambios drásticos en la mayoría de los casos. En si el hecho de usar CS se vio reflejado en mejora en tiempos de entrenamiento cuando se usó la arquitectura 1, aunque para ambos diseños se observó una obtención de patrones más efectiva ante las complejas series de tiempo.

Fue desarrollada la plataforma ETHAN el cual es un entorno de visualización de predicción de variables climáticas para los próximos 72 días de una ubicación en específico dado soporte gráfico, cuantitativo y cualitativo de los datos, la existencia de una API de la base de datos fue de gran importancia para realizar a ETHAN ya que se pudo obtener los datos cada día de manera autónoma y su respectiva predicción equiparando en su totalidad el funcionamiento completo de una plataforma de predicción del estado del tiempo, siendo lo más viable cuando no se cuenta con un equipo de adquisición de datos meteorológicos.

El modelo diseñado fue capaz de aprender las características meteorológicas de una segunda ubicación de forma óptima con las mismas limitaciones presentes en las pruebas iniciales. Demostrando que el desarrolló tiene características escalables para distintas condiciones térmicas. Manteniendo un nivel de tolerancia de acuerdo a la variabilidad del parámetro, variables como temperatura en la ubicación uno que variaban en una magnitud de 6ºC se tomó una tolerancia de ±2º C mientras que la humedad relativa de variable en magnitud de 35 % se eligió ±10% para una evaluación adecuada. Es necesario entrenar el sistema con las características de lugar dado porque cada sitio presenta un comportamiento distinto determinado no solo por el medio ambiente sino por la influencia del hombre en el mismo, además cada cierto tiempo se debe anexar nuevas mediciones que permitan al sistema estar al tanto y aprender los cambios presentes en los últimos tiempos.

Bibliografía

[1] M. Jiménez-Carrión, «Modelado y Predicción del Fenómeno El Niño en Piura, Perú mediante Redes Neuronales Artificiales usando Matlab,» *SciELO,* vol. 29, nº 4, 2018.

[2] M. P. G. Villegas, «Sistema multiagente para la predicción de clima para usos agrícolas,» *Revista I+D Tecnológico,* vol. 13, nº 1, 2017.

[3] F. Mercado, «Sistema de inteligencia artificial para la predicción temprana de heladas meteorológicas,» vol. 7, nº 4, 2016.

[4] I. Maqsood, «An ensemble of neural networks for weather forecasting, Neural Comput & Applic,» vol. 13, 2004.

[5] S. Mathur, «A Feature Based Neural Network Model for,» *Engineering and Technology,* vol. 34, 2007.

[6] M. Hayati, «Application of Artificial Neural Networks for Temperature Forecasting,» *Engineering and Technology ,* vol. 28, 2007.

[7] S. Baboo, «An eficient weather forecasting system using Altificial neural network,» *International Journal of Environmental Science and Development,* vol. 2, 2010.

[8] S. Caltagirone, «Air temperature prediction using evolutionary atfi cial neural networks,» vol. 12, 2001.

[9] «IDEAM,» [En línea]. Available: http://www.ideam.gov.co/web/tiempo-yclima/clima. [Último acceso: 11 mayo 2020].

[10] «Blogs.hoy.es,» 11 mayo 2020. [En línea]. Available: http://blogs.hoy.es/ciencia-facil/2013/02/11/que-es-y-como-funciona-el-sistema-climatico/?ref=https%3A%2F%2Fwww.google.com%2F.

[11] [En línea]. Available: http://www.ideam.gov.co/web/tiempo-y-clima/variabilidad-climatica. [Último acceso: 11 mayo 2020].

[12] J. Malpica, Departamento de Matemáticas de la UAH, [En línea]. Available: http://www3.uah.es/benito_fraile/ponencias/inteligencia-artificial.pdf.

[13] S. D. Warden P., Machine Learning with TensorFlow Lite on Arduino and Ultra-Low-Power Microcontrollers, 1 ed., M. Loukides, Ed., Sebastopol, California: O'Reilly Media, 2019.

[14] «mc.ai,» 5 Diciembre 2019. [En línea]. Available: https://mc.ai/its-deep-learning-times-a-new-frontier-of-data/. [Último acceso: 11 mayo 2020].

[15] «ambientech.org,» [En línea]. Available: https://ambientech.org/la-neurona. [Último acceso: 11 mayo 2020].

[16] F. Ucha, «DefinicióABC,» julio 2010. [En línea]. Available: https://www.definicionabc.com/ciencia/neurona.php. [Último acceso: 11 mayo 2020].

[17] I. Rovira, «Psicología y Mente,» [En línea]. Available: https://psicologiaymente.com/neurociencias/partes-de-neurona. [Último acceso: 11 mayo 2020].

[18] E. Zamora, «Curso de Redes Neuronales,» Ciudad de México, 2015.

[19] «bioquimicadental,» enero 2016. [En línea]. Available: https://bioquimicadental.wordpress.com/2016/01/21/potencial-de-accion/. [Último acceso: 11 mayo 2020].

[20] S. Raschka, «http://rasbt.github.io/,» 2019. [En línea]. Available: http://rasbt.github.io/mlxtend/user_guide/general_concepts/activation-functions/. [Último acceso: 11 mayo 2020].

[21] S. Verna, 20 junio 2019. [En línea]. Available: https://towardsdatascience.com/understanding-different-loss-functions-for-neural-networks-dd1ed0274718. [Último acceso: 11 mayo 2020].

[22] C. Vega, «¿Qué es una Red Neuronal? Parte 3.5 : Las Matemáticas de Backpropagation | DotCSV,» 2019.

[23] S. Doshi, «medium.com,» 13 enero 2019. [En línea]. Available: https://medium.com/@sdoshi579/optimizers-for-training-neural-network-59450d71caf6. [Último acceso: 11 mayo 2020].

[24] L. Sugar. [En línea]. Available: https://ccc.inaoep.mx/~esucar/Clases-MetIA/MetIA-16.pdf.

[25] M. Diaz. [En línea]. Available: http://bibing.us.es/proyectos/abreproy/11933/fichero/MEMORIA%252F2%29 Introducci%C3 %B3n+a+la+teor%C3%ADa+del+Compressive+Sensing.pdf. [Último acceso: 11 mayo 2020].

[26] Y. Sheng, 1996.

[27] D. O. N. Nieto, «EL USO DE LA TRANSFORMADA WAVELET DISCRETA EN LA RECONSTRUCCIÓN DE SEÑALES SENOSOIDALES,» *Scientia et Technica Año XIV,* nº 38, 2008.

[28] «http://catarina.udlap.mx/,» [En línea]. Available: http://catarina.udlap.mx/u_dl_a/tales/documentos/meie/osorio_s_a/capitulo2.pdf. [Último acceso: 11 mayo 2020].

[29] [En línea]. Available: http://www.dmae.upct.es/~paredes/am_ti/apuntes/Tema%202.%20Series%20y%20transformadas%20de%20Fourier.pdf. [Último acceso: 11 mayo 2020].

[30] E. Soria. [En línea]. Available: https://www.uv.es/soriae/tema_5_pds.pdf. [Último acceso: 11 mayo 2020].

[31] X. Grandio, 14 julio 2017. [En línea]. Available: https://marketing4ecommerce.net/tensorflow-que-es-y-sus-aplicaciones/. [Último acceso: 11 mayo 2020].

[32] J. Torres. [En línea]. Available: https://torres.ai/deep-learning-inteligencia-artificial-keras/. [Último acceso: 11 mayo 2020].

[33] [En línea]. Available: https://www.redhat.com/es/topics/api/what-are-application-programming-interfaces. [Último acceso: 11 mayo 2020].

[34] J. Ordoñez. [En línea]. Available: https://www.idento.es/blog/desarrollo-web/que-es-una-api-rest/. [Último acceso: 11 mayo 2020].

[35] P. Stackhousw, 11 febrero 2020. [En línea]. Available: https://power.larc.nasa.gov/docs/services/api/v1/. [Último acceso: 11 mayo 2020].

[36] G. Moreno. [En línea]. Available: http://documentacion.ideam.gov.co/openbiblio/bvirtual/009375/009375.pdf. [Último acceso: 11 mayo 2020].

ANEXO

Estos anexos describen las tablas para cada variable considerando las 3 longitudes de predicción y los porcentajes de precisión para los mismo, estas tablas son la base para las gráficas de barras de la sección de resultados.

Anexos i Resultados Temperatura con Arquitectura 1 Ubicación 1

Temperatura $\pm 2ºC$		Modelo 1 con CS			Modelo 1 sin CS			
Lp	Comp-CS	Prec (%)	RMSE	TE (min)	LS	Prec (%)	RMSE	TE (min)
1	60%	90.52	1.59	7.86	52	87.36	1.02	6.03
	70%	90.53	1.33	3.85	39	87.89	1.03	6.44
	80%	90.52	1.42	3.6	26	87.36	1.07	6.43
2	60%	91.05	1.37	4.6	52	94.21	1.02	7.1
	70%	94.73	1.14	6.7	39	93.15	1.08	9.46
	80%	93.15	1.16	6.06	26	94.21	1.00	4.55
5	60%	90.52	1.28	4.53	52	95.16	0.98	8.51
	70%	91.57	1.30	4.63	39	93.18	1.03	8.63
	80%	90	1.19	6.9	26	92.63	1.1	7.11
10	60%	92.63	1.16	2.6	52	91.57	0.98	7.2
	70%	92.63	1.14	2.66	39	92.63	1.04	7.33
	80%	93.68	1.11	3.75	26	94.73	0.97	6.38

Anexos ii Resultados Humedad Relativa con Arquitectura 1 Ubicación 1

Humedad Relativa $\pm 10\%$		Modelo 1 con CS			Modelo 1 sin CS			
Lp	Comp-CS	Prec (%)	RMSE	TE (min)	LS	Prec (%)	RMSE	TE (min)
1	60%	88.47	44.96	5.69	52	88.41	43.26	14.86
	70%	84.73	52.48	5.67	39	92.21	37.81	11.72
	80%	86.31	47.64	10.32	26	92.10	41.02	13.04
2	60%	87.3	47.93	8.35	52	93.68	39.78	14.06
	70%	85.78	52.94	6.76	39	93.15	40.37	13.8
	80%	90	46.12	5.4	26	92.10	38.71	23.26
5	60%	86.31	52.43	7.03	52	94.21	37.66	12.7
	70%	86.84	47.08	5,5	39	90.52	42.75	7.62
	80%	86.31	47.25	13.98	26	93.16	40.11	25.04
10	60%	88.94	40.88	10.48	52	92.63	38.72	13
	70%	90.52	45.73	11.75	39	92.10	35.53	12.88

	80%	90	41.08	9.1	26	92.63	37.06	24.12

Anexos iii Resultados Velocidad del Viento Arquitectura 1 Ubicación 1

Velocidad del Viento $\pm 1.5 km/h$		Modelo 1 con CS			Modelo 1 sin CS			
Lp	Comp-CS	Prec (%)	RMSE	TE (min)	LS	Prec (%)	RMSE	TE (min)
1	60%	84.73	1.25	7.75	52	86.84	1.04	7.47
	70%	82.63	1.23	6.73	39	88.90	1.03	6.88
	80%	84.73	1.13	6.07	26	87.89	0.95	5.88
2	60%	79.47	1.23	14.33	52	86.84	0.98	6,35
	70%	80	1.11	10.41	39	87.83	1.03	6.35
	80%	83.15	1.21	9.33	26	85.26	1.15	9.36
5	60%	80	1.32	18.75	52	81.57	1.15	6.71
	70%	84.21	1.16	19.06	39	85.78	1.03	6.23
	80%	85.26	1.24	18.81	26	84.73	1.01	9.08
10	60%	82.10	1.2	14.48	52	85.78	1.07	6.4
	70%	83.15	1.12	13.96	39	90	0.96	6.33
	80%	82.63	1.25	13.36	26	86.84	0.94	6.23

Anexos iv Resultados Precipitación con Arquitectura 1 Ubicación 1

Precipitación $\pm 10 mm/dia$		Modelo 1 con CS			Modelo 1 sin CS			
Lp	Comp-CS	Prec (%)	RMSE	TE (min)	LS	Prec (%)	RMSE	TE (min)
1	60%	77.89	94.71	7.30	52	85.26	82.13	6.05
	70%	80	110.06	7.31	39	85.28	79.93	7.58
	80%	78.41	93.01	3.96	26	82.10	90.69	13.27
2	60%	84.21	66.95	20.5	52	84.41	89.64	8.93
	70%	83.15	76.05	20.56	39	87.36	71.17	8.23
	80%	82.63	71.02	19.6	26	78.94	125.43	6.33
5	60%	84.21	72.75	15.45	52	86.63	88.60	10.31
	70%	84.73	72.92	20.25	39	84.21	80.39	11.21
	80%	84.21	84.40	21.13	26	78.42	92.39	11.6
10	60%	83.15	69.13	7.45	52	83.78	81.69	10.58
	70%	84.21	69.79	6.73	39	83.68	90.32	15.31
	80%	86.84	67.51	6.51	26	84.21	81.02	10.76

Anexos v Resultados Insolación con Arquitectura 1 Ubicación 1

Insolación $\pm 1.5\ wh/km2$		Modelo 1 con CS			Modelo 1 sin CS			
Lp	Comp-CS	Prec (%)	RMSE	TE (min)	LS	Prec (%)	RMSE	TE (min)
1	60%	80.52	1.59	7.31	52	86.84	1.16	12.87
	70%	78.94	1.59	7.39	39	80	1.20	15.06
	80%	78.94	1.45	4.15	26	73.15	1.48	9.46
2	60%	78.94	1.56	5.06	52	87.89	1.08	17.56
	70%	77.36	1.62	8.46	39	84.21	1.19	12,78
	80%	82.10	1.40	5.2	26	81.05	1.43	10.4
5	60%	80	1.37	5.46	52	86.31	1.12	15.43
	70%	77.89	1.51	5.75	39	87.89	1.05	20.56
	80%	81.57	1.23	7.23	26	84.21	1.14	14.98
10	60%	83.15	1.34	9.48	52	85.78	1.05	16.09
	70%	86.31	1.26	5.93	39	85.78	1.17	24.98
	80%	85.26	1.26	6.11	26	84.21	1.09	12.09

Anexos vi Resultados Temperatura con Arquitectura 2 Ubicación 1

Temperatura $\pm 2ºC$		Modelo 2 con CS			Modelo 2 sin CS			
Lp	Comp-CS	Prec (%)	RMSE	TE (min)	LS	Prec (%)	RMSE	TE (min)
1	60%	92.63	1.20	36.8	52	88.90	1.09	26.61
	70%	91.25	1.25	29.68	39	89.47	1.00	25.64
	80%	90	1.43	37.11	26	86.84	1.06	28,66
2	60%	92.26	1.17	22.11	52	93.68	1.06	30,45
	70%	92.10	1.53	24.93	39	93.15	1.13	40.56
	80%	89.47	1.44	30.1	26	93.15	1.04	45,87
5	60%	93.68	1.16	35.7	52	94.21	0.95	13.11
	70%	89.47	1.45	38.2	39	93.15	1.03	50.45
	80%	85.26	1.83	26.95	26	93.68	1.13	55,34
10	60%	90.52	1.41	3156	52	93.68	1.01	40.56
	70%	92.63	1.21	56.4	39	92.10	1.20	45.76
	80%	89.47	1.37	25.06	26	95.26	0.99	32.14

Anexos vii Resultados Humedad Relativa con Arquitectura 2 Ubicación 1

Humedad Relativa $\pm 10\%$		Modelo 2 con CS			Modelo 2 sin CS			
LP	Comp-CS	Prec (%)	RMSE	TE (min)	LS	Prec (%)	RMSE	TE (min)
1	60%	86.84	46.98	32.08	52	93.68	36.92	22.96
	70%	87.36	48.86	23.83	39	90	39.52	20.06
	80%	83.68	56.45	30.01	26	89.47	44.37	11.88
2	60%	88.42	43.39	43.08	52	92.63	41.87	18.86
	70%	83.15	53.84	31.38	39	90	42.86	22.13
	80%	83.62	56.78	23.96	26	87.36	49.69	22.45
5	60%	87.89	46.61	35.73	52	93.61	36.41	18.23
	70%	85.78	49.73	38.75	39	90.52	44.43	23.43
	80%	81.57	58.57	47.5	26	92.10	41.86	22.12
10	60%	88.94	42.76	37.11	52	89.47	43.46	18.4
	70%	87.36	38.68	37.45	39	93.68	43.67	18.43
	80%	84.73	58.62	78.83	26	86.84	44.30	22.45

Anexos viii Resultados Velocidad del Viento con Arquitectura 2 Ubicación 1

Velocidad del Viento $\pm 1.5 km/h$		Modelo 2 con CS			Modelo 2 sin CS			
LP	Comp-CS	Prec (%)	RMSE	TE (min)	LS	Prec (%)	RMSE	TE (min)
1	60%	83.68	1.11	36.74	52	85.78	1.06	18.35
	70%	81.05	1.27	36.74	39	84.73	1.08	18.54
	80%	88.94	0.86	30.41	26	83.68	1.06	22.45
2	60%	80.52	1.26	32.93	52	86.84	1.02	22.56
	70%	76.31	1.42	33.52	39	85.78	1.08	18.23
	80%	81.57	1.31	63.06	26	84.73	1.13	32.45
5	60%	81.05	1.24	28.78	52	87.36	1.08	18.54
	70%	81.57	1.35	24.55	39	86.84	1.10	18.43
	80%	81.05	1.30	27.43	26	84.21	1.06	19.43
10	60%	82.10	1.12	35.55	52	84.21	1.14	19.45
	70%	85.78	1.09	30.72	39	84.21	1.18	25.05
	80%	80.52	1.43	33.05	26	87.36	1.02	12.05

Anexos ix Resultados Precipitación con Arquitectura 2 Ubicación 1

Precipitación $\pm 10\%$		Modelo 2 con CS			Modelo 2 sin CS			
LP	Comp-CS	Prec (%)	RMSE	TE (min)	LS	Prec (%)	RMSE	TE (min)
1	60%	77.98	114,96	26,42	52	81.57	90.79	10.26
	70%	74.21	106.47	33.36	39	60.25	125.92	10.37
	80%	79.47	110.17	34.13	26	80	92.94	10.03
2	60%	80.52	79.67	65.98	52	84.21	80.02	18.16
	70%	83.15	80.02	35.11	39	86.31	74.92	12.24
	80%	88.42	63.80	53.73	26	86.38	86.13	18.21
5	60%	84.73	69.26	76.96	52	81.57	93.10	18.23
	70%	85.26	62.27	33.3	39	81.05	84.71	18.11
	80%	75.78	95.20	46.96	26	86.38	86.91	18.35
10	60%	85.26	64.13	34.4	52	85.26	87.52	26.53
	70%	83.68	85.21	28.48	39	85.26	84.42	18.33
	80%	80.52	84.37	48.55	26	84.73	80.27	18.58

Anexos x Resultados Insolación con Arquitectura 2 Ubicación 1

Insolación $\pm 1.5\ wh/km2$		Modelo 2 con CS			Modelo 2 sin CS			
LP	Comp-CS	Prec (%)	RMSE	TE (min)	LS	Prec (%)	RMSE	TE (min)
1	60%	73.68	1.87	28.78	52	84.73	1.13	12.25
	70%	77.36	1.41	26.29	39	85.78	1.22	10.69
	80%	81.05	1.45	15.12	26	86.31	1.17	10.70
2	60%	78.94	1.44	27.73	52	88.94	1.14	19.06
	70%	83.68	1.30	29.3	39	84.21	1.23	20.64
	80%	74.73	1.51	26.08	26	81.57	1.18	20.8
5	60%	77.36	1.43	28.08	52	84.21	1.24	17.76
	70%	80.52	1.44	24.47	39	86.84	1.16	21.39
	80%	81.05	1.38	31.06	26	85.26	1.15	14.18
10	60%	82.10	1.37	31.7	52	83.68	1.15	25.44
	70%	82.10	1.40	29.5	39	84.21	1.24	28.31
	80%	82.63	1.56	27.73	26	85.78	1.16	22.71

Anexos xi Estadística de Predicción de Temperatura en ubicación 2

Temperatura $\pm 1.5\ ^{a}C$		Modelo 1 con CS		
LP	Comp-CS	Prec (%)	RMSE	TE (min)
1	60%	74.73	1.59	8.50
	70%	79.47	1.47	8.83
	80%	74.73	1.46	10.05
2	60%	84.73	1.12	8.84
	70%	80.52	1.24	8.48
	80%	81.05	1.45	9.51

Anexos xii Estadísticas de Predicción de Humedad Relativa Ubicación 2

Humedad Relativa ± 5 %		Modelo 1 con CS		
LP	Comp-CS	Prec (%)	RMSE	TE (min)
1	60%	80	16.79	8.81
	70%	83.15	15.12	8.71
	80%	80.52	14.37	9.39
2	60%	83.115	14.70	8.64
	70%	83.68	14.62	8.64
	80%	84.21	13.78	9.14

Anexos xiii Estadística de Predicción de Precipitación Ubicación 2

Precipitación $\pm 2\ mm/dia$		Modelo 2 con CS		
LP	Comp-CS	Prec (%)	RMSE	TE (min)
1	60%	76.84	16.21	8.22
	70%	74.21	16.88	8.81
	80%	76.84	12.58	9.71
2	60%	71.57	14.20	4.76
	70%	67.89	18.64	8.93
	80%	72.63	11.21	16.88

Anexos xiv Estadísticas de Predicción de Insolación Ubicación 2

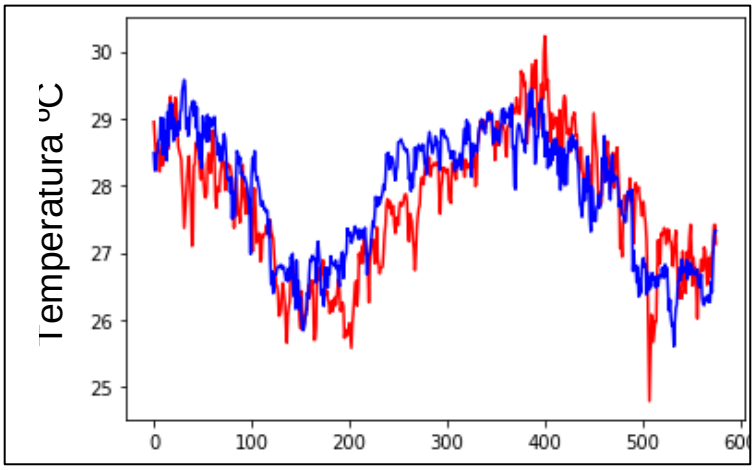
Insolación $\pm 2\ kwh/m2$		Modelo 2 con CS		
LP	Comp-CS	Prec (%)	RMSE	TE (min)
1	60%	79.47	2.76	7.78
	70%	83.15	2.48	7.77
	80%	81.57	2.62	13.34
2	60%	77.36	2.85	8.72
	70%	77.89	2.92	9.63
	80%	78.94	2.63	16.16

Anexos xv Estadísticas de Predicción de Velocidad del Viento Ubicación 2

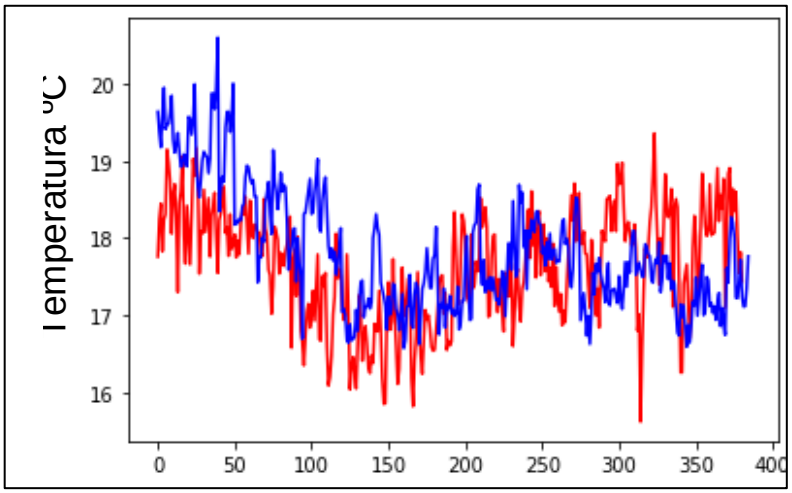
Velocidad del Viento $\pm 2\ m/s$		Modelo 2 con CS		
LP	Comp-CS	Prec (%)	RMSE	TE (min)
1	60%	68.41	4.15	8.19
	70%	71.05	4.18	8.86
	80%	57.36	5.08	11.53
2	60%	62.10	5.25	8.87
	70%	63.68	4.14	8.94
	80%	68.94	3.96	6.72

En esta sección se ilustra una actualización de los modelos de predicción ajustando la arquitectura 1, adecuando la Matriz compresión en una zona que permitía establecer patrones distintos de una mejor manera. Estos casos usan una longitud de predicción de 5. Aumentando el periodo de evaluación de 190 a 576 días.

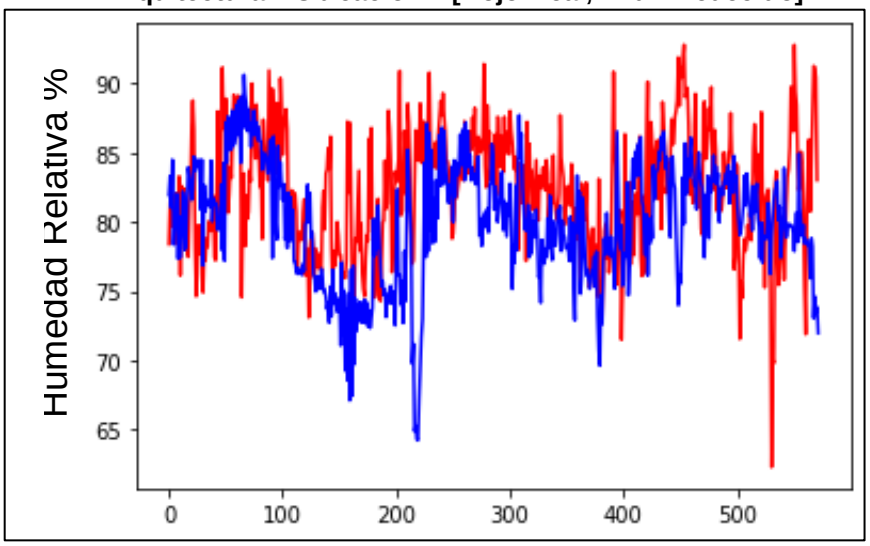
Anexos xvi Modelo de Predicción para Temperatura Mejorado 70 % Compresión Arquitectura 1 Ubicación 2 [Rojo-Real, Azul-Predecido]



Anexos xvii Modelo de Predicción para Temperatura Mejorado 70 % Compresión Arquitectura 1 Ubicación 1 [Rojo-Real, Azul-Predecido]



Anexos xviii Modelo de Predicción para Humedad Relativa Mejorado 70 % Compresión Arquitectura 1 Ubicación 1 [Rojo-Real, Azul-Predecido]



Anexos xix Modelo de Predicción para Velocidad del Viento Mejorado 70 % Compresión Arquitectura 1 Ubicación 2[Rojo-Real, Azul-Predecido]