UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE CIENCIAS

Sistema de Redes Neuronales para la predicción de ozono en la CDMX y el área metropolitana.

T E S I S

QUE PARA OBTENER EL TÍTULO DE:

Licenciado en Ciencias de la Computación

PRESENTA:

Pablo Camacho González

TUTOR

Dr. Olmo S. Zavala Romero

Ciudad Universitaria, Cd. Mx., 2018

Universidad Nacional Autónoma de México
UNAM
Biblioteca Central
Dirección General de Bibliotecas de la UNAM

UNAM – Dirección General de Bibliotecas
Tesis Digitales
Restricciones de uso

DERECHOS RESERVADOS ©
PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL

Todo el material contenido en esta tesis esta protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de imágenes, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente donde la obtuvo mencionando el autor o autores. Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.

Agradecimientos.

A mis padres, porque sin ellos nada de esto sería posible.

A mis hermanos, por compartir gran parte de su vida conmigo.

A todos los amigos y compañeros con los cuales en algún momento de mi vida compartí momentos que son parte de quien soy.

Al Dr. Olmo Zavala Romero, por aceptarme como su estudiante y todo lo que esto involucra.

Al grupo IOA del Centro de Ciencias de la Atmósfera, porque esta tesis no sería posible sin el trabajo que hacen todos los días.

A Pavel Ernesto Oropeza Alfaro y M. en C. Dulce Herrera Moro, por la ayuda en la configuración y el mantenimiento del servidor de desarrollo Zion, donde se desarrolló el trabajo escrito en la presente tesis.

Al Programa de Apoyo a Proyectos de Investigación e Innovación Tecnológica (PAPIIT IA103917), por la beca otorgada para desarrollar la investigación que dio como resultado esta tesis, también al Sistema de pronostico Meteorológico y de Calidad del aíre (SECITI 120/2017)

Índice general

Índice de figuras

Introducción

Noviembre de 1986, Ciudad de México. La ciudad era gris de nuevo, la fiesta por haber albergado un mundial de fútbol se acabó, sólo era un recuerdo. No quedó más que regresar a la rutina, esa rutina donde cientos de automóviles recorren las avenidas, cientos de personas trabajan en las grandes fábricas, esa rutina en la que una masa de aire gris cubre la ciudad. 25 de noviembre, el día más contaminado en la historia de esta ciudad, 491 puntos en el Índice Metropolitano de la Calidad de Aire (IMECA), la inminente crisis ambiental quedaba al descubierto. Las causas, la falta de verificación de los automóviles, la baja calidad de los combustibles y la poca conciencia y conocimiento tanto de los habitantes como del gobierno [14] [16]. El efecto, los habitantes inhalan una gran cantidad de contaminantes tóxicos, poniendo en peligro su salud.

Al poner al descubierto esta problemática, el gobierno se vio acorralado, algo se tenía que hacer, las soluciones tenían que ser rápidas y eficaces. Un programa de verificación voluntaria de los automóviles, mejorar la calidad del combustible y la creación del precursor del "Hoy no circula" al que se le llamó "un día sin auto", fueron las soluciones propuestas.

En 1986 la Ciudad de México fue considerada una de las ciudades más contaminadas del planeta, después de este año, la ciudad ha tenido que conllevar este problema. En los años 2002 y 2016 el índice de la calidad del aire llegó a rebasar los 150 puntos,

poniendo en peligro la salud de sus habitantes una vez más. La actual Ciudad de México y el área metropolitana, habitada por cerca de 23 millones de habitantes [17], con un constante crecimiento en el uso de los automóviles y una localización geográfica cercana a zonas industriales, ha provocado que la contaminación atmosférica siga siendo un problema que se debe atacar para disminuir y prevenir los problemas que provoca. Una forma de mejorar la toma de decisiones para evitar que los niveles de contaminación sean tan elevados es el uso de pronósticos de calidad del aire. Los pronósticos de calidad del aire, o de la contaminación, están basados en la química, física y la dinámica de la atmósfera, y en la CDMX se cuenta con un pronóstico de calidad de aire desde el 2017 [33], ajustado a la dinámica atmosférica propia de la ciudad.

En las ciencias de la computación se tiene el área del *aprendizaje automático*, en la cual más que entender el cerebro, se enfoca en el desarrollo de algoritmos que le dan a una computadora la capacidad de aprender. Las redes neuronales artificiales, una de las técnicas más usadas en años recientes, busca imitar el comportamiento y la arquitectura de las neuronas naturales para dotar a una computadora con la capacidad de aprender [8], para nuestro problema el hecho de que una red neuronal aprenda se refiere a la capacidad de la red neuronal en dar un pronostico de ozono lo mas cercano al valor de ozono observado.

En la actualidad la contaminación atmosférica es uno de los problemas ambientales más preocupantes para las grandes megalópolis ya que el uso indiscriminado del automóvil, la numerosa cantidad de fabricas aunado a la poca conciencia y educación en temas ambientales de las personas, a propiciado problemas en la salud de los habitantes de las ciudades [32]. Los pronósticos de calidad del aire predicen múltiples contaminantes, entre los que destacan: Ozono, Monóxido de carbono, Dióxido de azufre y Dióxido de nitrógeno. De esta manera se puede alertar a la población o iniciar una

contingencia ambiental, una contingencia ambiental se inicia cuando hay altos niveles contaminación en la atmósfera que sean dañinos para la salud y consiste en tomar medidas temporales que pueden mejorar la calidad del aire.

En este trabajo se desarrolló un modelo para la predicción del índice de ozono a 24 horas, basado en un sistema de redes neuronales artificiales. La razón de por que se decidió usar las redes neuronales para la predicción de ozono, se debe al creciente uso de esta técnica para resolver problemas que no son lineales, dado que el problema de la predicción de ozono cae en esta categoría, resulta de interés saber si la redes neuronales son capaces de resolver este problema, ademas de buscar técnicas diferentes a las que se usan de manera convencional para resolver este problema.

Las redes neuronales son entrenadas con la información de las estaciones de la Red Automática de Monitoreo Atmosférico (RAMA), de la Secretaría del Medio Ambiente de la Ciudad de México, así como con información meteorológica de los últimos 7 años. Los datos meteorológicos se obtuvieron de las salidas del modelo para el pronóstico y la investigación del clima (WRF por sus siglas en inglés) hecho por el grupo de Interacción Océano-Atmósfera (IOA) del Centro de Ciencias de las Atmósfera (CCA) de la UNAM.

1. Ciudad de México y la contaminación atmosférica

Desde los 80's, la contaminación atmosférica se ha vuelto un problema constante para la CDMX, la zona geográfica en la que se encuentra, la cantidad de industrias que aportan algún contaminante a la atmósfera y el excesivo uso de combustibles fósiles, son algunas de las fuentes de contaminación con las que tiene que lidiar la ciudad. Para prevenir los altos índices de contaminantes, la ciudad de México cuenta con un un pronóstico de la calidad del aire y un Índice Metropolitano de la Calidad de Aire (IME-

CA). Con estos dos elementos, se busca informar y prevenir a la población. Además, se tienen programas como el "Hoy no circula" y la verificación obligatoria de los automóviles, los cuales se han visto rebasados por los cerca de 23 millones de habitantes y los casi 6 millones de vehículos automotores que habitan y circulan en la ciudad de México y el área metropolitana [17] [18].

Junto con los programas antes mencionados, la ciudad de México cuenta con dos fases de contingencia para tratar de reducir los días que rebasan altos índices de contaminación. Desde 2016, la primer fase se activa cuando el índice de ozono alcanza las 150 partes por millón (ppm). Mientras que la segunda fase se activa cuando se rebasa las 200 ppm. El Índice Metropolitano de la Calidad del Aire se calcula a partir de los valores observado de los contaminantes como son el ozono, PM_{10} y $PM_{2,5}$ y la ecuación 1 descrita más adelante, los contaminantes PM_{10} y $PM_{2,5}$ son partículas de polvo, cenizas, hollín, partículas metálicas, cemento o polen que se encuentran dispersas en la atmósfera y que su tamaño es menor a 10 y 2.5 micrómetros. Cuando el índice está por arriba de las 150 ppm se levanta una contingencia ambiental. El uso del pronóstico de la calidad del aire de la CDMX, en este momento, sólo es con el fin de informar a la población sobre el posible porvenir de la calidad del aire y que de esta manera la población pueda tomar precauciones [33].

El IMECA, es un indicador usado en la Ciudad de México para informar a la población el nivel de contaminación atmosférica y los riesgos que esta tiene sobre su salud. El indicador está basado en un cálculo sencillo, a partir de 2 puntos de quiebre; los puntos de quiebre son los valores conocidos en los cuales la fisiología de la población puede verse afectada. En la tabla 1 se muestran algunos rangos del IMECA y su relación con otros contaminantes [7]:

El cálculo del índice se hace con las fórmulas de la ecuación 1 y las tablas que se

IMECA	**O3** ppm	**PM10** $\frac{\mu g}{m^3}$	**PM2.5** $\frac{\mu g}{m^3}$
0 -50	0-70	0-40	0.0 - 12.0
51- 100	71 - 95	41 - 75	12.1 - 45.0
101 - 150	96 - 154	76 - 214	45.1 - 97.4
201 - 200	155 - 204	215 - 354	97.5 - 150.4
201 - 300	205 - 404	355 - 424	150.5 - 250.4
301 - 400	405 - 504	425 - 505	250.5 - 350.4
401 - 500	505 - 604	505 - 604	350.5 - 500.4

Tabla 1: Tabla de puntos de quiebre para los contaminantes O_3, PM_{10} y $PM_{2,5}$

muestran en la figura 1:

$$\begin{aligned} \text{indice} &= (k * (C - BP_{LO})) + I_{LO} \\ k &= \frac{I_{Hi} - I_{lo}}{BP_{Hi} - BP_{LO}} \end{aligned} \tag{1}$$

En donde:

- C = valor redondeado para la concentración del contaminante deseado (O_3, PM_{10} o $PM_{2,5}$).
- k = constante de proporcionalidad estimada.
- BP_{Hi} = valor del punto de corte que es mayor o igual a la concentración a evaluar.
- BP_{LO} = valor del punto de corte que es menor o igual a la concentración a evaluar.
- I_{Hi} = valor de índice de punto de corte que corresponde al valor de BP_{Hi}.
- I_{LO} = valor de índice de punto de corte que corresponde al valor de BP_{LO}.

Concentración de O_3 (Promedio de 1h, ppm)	Concentraciones para los puntos de corte (ppm)		Equivalencia en el índice para los puntos de corte		k	Categoría
	BP_{Hi}	BP_{Lo}	I_{Hi}	I_{Lo}		
0.000 – 0.070	0.07	0	50	0	714.29	BUENA
0.071 – 0.095	0.095	0.071	100	51	2041.67	REGULAR
0.096 – 0.154	0.154	0.096	150	101	844.83	MALA
0.155 – 0.204	0.204	0.155	200	151	1000.00	MUY MALA
0.205 – 0.404	0.404	0.205	300	201	497.49	EXTREMADAMENTE MALA
0.405 – 0.504	0.504	0.405	400	301	1000.00	
0.505 – 0.604	0.604	0.505	500	401	1000.00	

Figura 1: Equivalencias para ozono (O_3), tabla tomada de la pagina http://www.aire.cdmx.gob.mx

Concentración de PM_{10} (Promedio móvil de 24h)	Concentraciones para los puntos de corte (ppm)		Equivalencia en el índice para los puntos de corte		k	Categoría
	BP_{Hi}	BP_{Lo}	I_{Hi}	I_{Lo}		
0 – 40	40	0	50	0	1.2500	BUENA
41 – 75	75	41	100	51	1.4412	REGULAR
76 – 214	214	76	150	101	0.3551	MALA
215 – 354	354	215	200	151	0.3525	MUY MALA
355 – 424	424	355	300	201	1.4348	EXTREMADAMENTE MALA
425 – 504	504	425	400	301	1.2532	
505 – 604	604	505	500	401	1.0000	

Concentración de $PM_{2.5}$ Â (Promedio móvil de 24h)	Concentraciones para los puntos de corte (ppm)		Equivalencia en el índice para los puntos de corte		k	Categoría
	BP_{Hi}	BP_{Lo}	I_{Hi}	I_{Lo}		
0.0 – 12.0	12	0	50	0	4.1667	BUENA
12.1 – 45.0	45	12.1	100	51	1.4894	REGULAR
45.1 – 97.4	97.4	45.1	150	101	0.9369	MALA
97.5 – 150.4	150.4	97.5	200	151	0.9263	MUY MALA
150.5 – 250.4	250.4	150.5	300	201	0.9910	EXTREMADAMENTE MALA
250.5 – 350.4	350.4	250.5	400	301	0.9910	
350.5 – 500.4	500.4	350.5	500	401	0.6604	

Figura 2: Equivalencias para Partículas Menores a 10 micrómetros (PM_{10}) y 2.5 micrómetros ($PM_{2,5}$), tabla tomada de la pagina http://www.aire.cdmx.gob.mx

2. Modelos de la calidad del aire

Los modelos de la calidad de aire están divididos en 4 tipos: gaussianos, numéricos, estadísticos y físicos. Esta división se basa en el conocimiento de los elementos que influyen de manera positiva o negativa en la contaminación atmosférica.

Los modelos **gaussianos**, son usados principalmente para medir el impacto de los contaminantes que no son reactivos. Los modelos **numéricos** son usados cuando se quiere medir la contaminación proveniente de múltiples fuentes de contaminantes, este tipo de modelos de la calidad del aire comúnmente cuenta con sistemas internos para la simulación de la emisión, transporte, difusión, transformación y remoción de los contaminantes, además de que necesitan de una gran cantidad de datos y un alto poder de cómputo para que den buenos resultados [36]. Los modelos **estadísticos**, usualmente son usados cuando no se tiene suficiente poder de cómputo o cuando los niveles de contaminación se pueden modelar bien a partir de datos históricos, mientras que los modelos **físicos** requieren de un gran conocimiento en técnicas de modelado de fluidos para poder medir el impacto de la calidad de aire en una área determinada.

Actualmente el modelo que usa la CDMX es el CMAQ (Community Multi-scale Air Quality), el cual es del tipo numérico. En específico es un modelo euleriano, y es desarrollado por la Agencia de Protección de Medio ambiente de los Estados Unidos (USEPA). [33] En el Centro de Ciencias de la Atmósfera (CCA) de la UNAM desde el 2000 se hace uso del modelo Multiscale Climate and Chemistry Model (MCCM) y a partir del 2010 del Weather Research and Forecasting (WRF-CHEM).

2.1. Modelos de dispersión

Los modelos de dispersión que pertenecen a los modelos gaussianos están construidos para predecir una situación de desplazamientos de gases y partículas, provenientes de diferentes fuentes existentes en el entorno estudiado. Los modelos de dispersión trabajan bajo series de tiempo, usualmente la información con la que funcionan estos modelos es la siguiente:

- Información meteorológica.

- Tasa de la emisión del contaminante.
- Velocidad y dirección del viento.
- Tipo de suelo.
- Topografía del área.

La emisión de los contaminantes se pueden observar por medio de series de curvas, esto permite describir su comportamiento mediante la distribución de Gauss (ecuación 2). De esta manera tenemos una ecuación donde la distribución ayudará a encontrar la concentración de los contaminantes.

$$\Phi_{\mu\sigma 2}(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{\frac{(u-\mu)^2}{2\sigma^2}} du, x \varepsilon \mathbb{R} \tag{2}$$

Un modelo euleriano que pertenece a modelos de tipo numéricos, describe el movimiento y la química de los contaminantes. Este tipo de modelo es usado cuando se tiene un patrón de emisión complejo o cuando se tiene que la transformación química de los contaminante es de gran importancia en la generación y el transporte de contaminantes. Para este modelo se necesita tener información sobre el inventario de emisiones, la calidad del aire y la meteorología; así como dividir en celdas la región en donde se usará el modelo.

Para resolver los sistema de ecuaciones que se usan en los modelos eulerianos se usan diferentes metodologías, entre las más sencillas se encuentra el método euleriano inverso, que se considera un método estable, pero costoso en términos computacionales [36].

2.2. Community Multi-scale Air Quality (CMAQ)

La primer versión del modelo CMAQ aparece en 1998 [1] y es usado principalmente por investigadores en el área de la calidad del aire. La primera generación del modelo se basa en la simulación de calidad del aire usando la química a escala local, así como la formula gaussiana usada en los modelos gaussianos (ecuación 2). Para la segunda generación, el modelo estaba en una escala regional. Por lo tanto, se necesitaba de un modelo capaz de modelar un mayor número de contaminantes, ya que existen diferentes fuentes de emisión las cuales contribuyen a los niveles de más de un solo contaminante. En este caso es importante tomar en cuenta los diferentes escenarios de liberación de contaminantes, ya que pueden ser desde un fenómeno localizado en un corto plazo hasta uno que cubra una gran región y de largo plazo.

Actualmente CMAQ se encuentra en su tercera generación y se trata de un sistema tridimensional euleriano, trabaja con un modelo de química atmosférica; un modelo de transporte que simula el ozono, contaminantes tóxicos, la visibilidad y las especies de contaminantes ácidos, así como los nutrientes que se encuentran en la troposfera; finalmente un modelo matemático que simula la química y la física de la calidad del aire. Actualmente CMAQ se encuentra en su versión 5.02 y es usado por investigadores, gobierno, consultores, sector privado y la academia en más de 50 países [1].

3. Aprendizaje Automático

El aprendizaje automático o *machine learning*, como se le conoce en inglés, es un campo de las ciencias computacionales que se enfoca en el desarrollo de algoritmos o técnicas para que una computadora sea capaz de “aprender”. Es decir, a través de un cierto número de datos que se le suministran a la computadora, ésta sea capaz de reconocer información similar o generar nuevo conocimiento a partir de los datos de en-

trada. Existen dos tipos principales de aprendizaje automático: aprendizaje supervisado y aprendizaje no supervisado.

3.1. Aprendizaje no supervisado

Los algoritmos de aprendizaje no supervisado, son algoritmos de agrupamiento, que reciben ejemplos sin información sobre el grupo al que pertenecen. El algoritmo se encargará de poner a cada ejemplo en el grupo al que posiblemente puede pertenecer.

Una de las principales técnicas usadas en el aprendizaje no supervisado es K-medias. K-medias es una técnica de agrupamiento que tiene como fin la partición de un conjunto con n elementos divididos en k particiones.

3.2. Aprendizaje supervisado

En el aprendizaje supervisado los ejemplos están formados por dos partes: una son las características o atributos y la segunda es el grupo al que pertenecen o un valor que es consecuencia de las características. Con esto en mente, supongamos que queremos un algoritmo de aprendizaje supervisado que se a capaz de reconocer a qué árbol pertenece una hoja, lo único que hay que hacer es darle ejemplos de hojas de diferentes árboles con sus pertinentes características, además de agregarle una etiqueta a qué árbol o planta le pertenece. Es como si fuera una tabla con dos columnas principales (tabla: 2), la primer columna son las características y la segunda, la etiqueta a la que pertenece.

Con esta información el algoritmo es capaz de aprender a clasificar hojas. Cuando quiera clasificar una hoja nueva, sólo faltará darle las características al algoritmo y éste me dirá a que árbol le pertenece.

Algunas de las principales técnicas usadas en aprendizaje supervisado son:

Características						**Etiqueta**
Forma	*Limbo*	*Tallo*	*Borde*	*Nervadura*	*Duracion*	*Árbol al que pertenece*
Corazón	Ovaladas	Aislada	Dentado	Radial	Perennes	Árbol 1
Elíptica	Lanceoladas	Verticiladas	Hendida	Palmeada	Marcescentes	Árbol 2

Tabla 2: Ejemplos de datos etiquetados, con características y etiquetas, para algoritmos supervisados.

- **Redes Neuronales artificiales**. Este conjunto de algoritmos está basado en el comportamiento de una red neuronal biológica y está dividido en dos partes principales. El primero llamado propagación hacia adelante (*feedforward*) y el segundo propagación hacia atrás (*backpropagation*) [12]. Esta técnica se describe en la sección 4.2.

- **Regresión lineal**. Algoritmos que aproximan una función de forma lineal para ajustar a los datos de entrenamiento. También se pueden adaptar para que los algoritmos aproximen una función polinomial.

- **Máquinas de vectores de soporte (SVM)**. Los algoritmos SVM son usados principalmente en problemas de clasificación, para este algoritmo los datos se representan como puntos en un espacio, de tal forma que cada punto está etiquetado con la clase a la cual pertenece.

  La forma en que SVM encuentra una clasificación es: dado un subconjunto de datos de entrenamiento que ya se sabe a qué clase pertenecen, el algoritmo busca una recta, de tal forma que la recta divide a los datos de entrenamiento en su pertinente clase, la recta encontrada por el algoritmo tiene que cumplir que la distancia que hay entre los puntos más cercanos a la recta se la mayor posible. Cuando un nuevo ejemplo llega al algoritmo este es clasificado según su posición en el espacio y con respecto a la recta que el algoritmo encontró.

En este trabajo se utilizan redes neuronales debido a la gran capacidad de predecir

y clasificar patrones.

4. Redes neuronales artificiales

4.1. Historia

En los inicios del desarrollo de la redes neuronales artificiales el funcionamiento de una red biológica ya estaba fundamentado en la doctrina de la **neurona** [9] . La doctrina ve a las neuronas como entes discretos que al interactuar forman una red neuronal. Además del conocimiento de la doctrina, también se tenía el conocimientos de la **sinapsis**, la sinapsis es el proceso que ocurre para establecer la conexión de dos neuronas. Para 1932, 11 años antes de las redes neuronales artificiales, Edgar Douglas Adrian ya había registrado los **potenciales de acción** [11], estos se relacionan con el **umbral** de una neurona, el umbral se puede ver como un límite, cuando este se pasa, la neurona se conecta con otras neuronas. La interconexión de las neuronas no es al azar, la conexión de neurona y neurona es específica en puntos de contactos concretos, este hecho fue descrito en el principio de la especificidad de conexión [9].

Para 1943, fecha en que se inició el desarrollo de las redes neuronales artificiales, el funcionamiento de su contra-parte ya se tenía claro. Es por eso que el desarrollo de la redes artificiales está basada en las biológicas, tomando a la neurona como unidad mínima, la sinapsis como modelo de conexión de la función de activación con el perceptrón y el potencial de acción junto con el umbral como modelo de la función de activación. A pesar de tomar como base el funcionamiento de la redes biológicas, la redes neuronales artificiales están basadas en el cálculo, el álgebra, la estadística y la probabilidad, además de contar con un algoritmo llamado propagación hacia atrás.

Cuando Warren McCulloch y Walter Pitts [36] desarrollaron el primer modelo de

una red neuronal basado en un modelo matemático y un algoritmo, al que se conoce como lógica del umbral, el desarrollo del modelo dividió el futuro de la investigación en dos ramas. La primera rama se enfoca en los procesos biológicos del cerebro, mientras que la segunda rama se enfoca en seguir desarrollando y mejorando las redes neuronales artificiales.

Para 1958, Frank Rosenblat propone un modelo matemático del núcleo de una neurona artificial. A éste modelo se le conoce como perceptrón y es capaz de usarse para simular circuitos básicos como el AND y el OR. Sin embargo, circuitos como la OR-exclusiva no es posible que una neurona lo simule. En 1969, la investigación hecha por Marvin Minsky y Seymour Papert [28] describe que el perceptrón básico no es suficiente para procesar la OR-exclusiva y plantea los problemas que se tienen en cuanto a la capacidad de procesamiento en los equipos de computo de esos años.

Después de la investigación hecha por Minsky y Papert el desarrollo de las redes neuronales quedó estancado por un tiempo, hasta 1975 cuando Paul Werbos desarrolló el algoritmo de propagación hacia atrás, siendo el algoritmo más importante para una red neuronal. Este algoritmo es fundamental en el funcionamiento de la redes neuronales con más de una sola capa de neuronas. Durante los años siguientes, el uso de las máquinas de soporte vectorial (SVM por sus siglas en inglés) y los algoritmos más simples como la regresión lineal obtuvieron mayor popularidad que las redes neuronales. A finales de los años 90's las redes neuronales regresaron al panorama de la inteligencia artificial, las razones principales fueron la mejora en la capacidad de procesamiento de las computadoras y la gran cantidad de datos en múltiples áreas, lo que se conoce como "Big Data". Todo esto han provocado que las redes neuronales se encuentren en lo que hoy se conoce como "Aprendizaje profundo".

4.2. Funcionamiento

Las redes neuronales artificiales se inician tomando como base un modelo sencillo de una red neuronal biológica. Una red neuronal biológica está formada por neuronas: la unidad más pequeña que procesa información en el cerebro. Las neuronas tienen entradas llamadas **dendritas**, las cuales reciben pulsos eléctricos que son procesados por el **núcleo** de la neurona. El núcleo genera corrientes eléctricas que salen por la parte de la neurona llamada **axón**. Por lo regular, el axón de la neurona se conecta con otras neuronas, formando así una red neuronal [20]. La figura 3 muestra un esquema gráfico del modelo de una neurona.

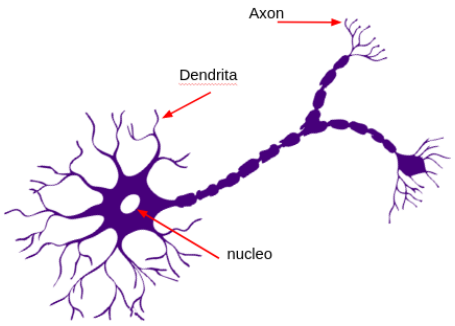

Figura 3: Partes de una neurona biológica

Cada neurona artificial contiene entradas (dendritas) que se representan como pesos, una función integradora de las entradas a la neurona llamada perceptrón (núcleo) y una función de activación o salida (axón). La figura 4 muestra de forma esquemática el modelo de una neurona artificial.

Las x_i representan neuronas conectadas a través de pesos w_i. La neurona o perceptrón integra las salidas de sus neuronas de entrada con la siguiente ecuación:

$$f(x) = w_1x_1 + w_2x_2 + w_2x_2... + w_nx_n \tag{3}$$

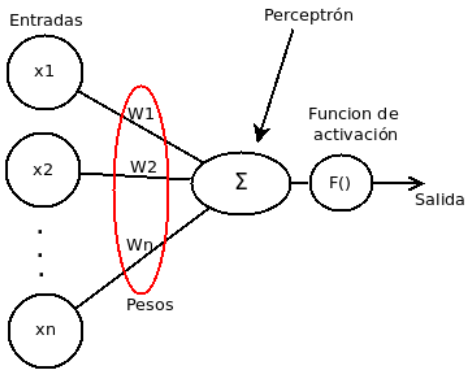

Figura 4: Modelo de una neurona en una red neuronal artificial.

La salida de la neurona se representa con una función de activación, comúnmente con una función **sigmoide**. La sección 4.3 describe a detalle las funciones de activación más comunes.

Una **red neuronal artificial** está conformada por múltiples neuronas que se agrupan de distintas maneras. La combinación del número de neuronas, el tipo de función de activación, el número de conexiones que puede tener una neurona con otras y el número de capas ocultas, da paso a tener un gran número de arquitecturas de redes neuronales. Cada una de las arquitecturas está enfocada en resolver un problema específico.

La mayoría de las arquitecturas de redes neuronales cuenta con una capa de entrada y una capa de salida. La diferencia se da en el perceptrón de cada neurona. El perceptrón puede contener un cálculo de probabilidad, de memoria o un cálculo de convolución. Otra diferencia entre cada arquitectura es la topología. La topología de una red neuronal consiste en la organización y la disposición de las neuronas para formar capas. Para la topología de una red neuronal hay que definir: el número de capas, el número de neuronas en cada capa, el grado de conectividad y el tipo de conexiones entre neuronas. Dentro de las diferentes topologías que puede tener una red neuronal existen tres principales: las de conexión hacia adelante, las de conexiones laterales, y

las de topología recurrente (figura 5).

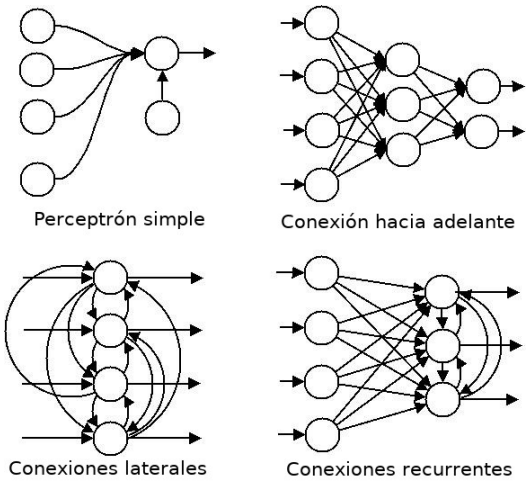

Figura 5: Diferentes tipos de topologías en una red neuronal [34] .

Una de las arquitecturas más utilizadas es la **perceptrón multicapa** que tiene una topología de conexión hacia adelante y agrupa conjuntos de neuronas en capas. La capa que recibe los valores de los atributos de nuestro problema se le denomina **capa de entrada**. La capa que contiene las salidas se llama **capa de salida** y el resto de las capas se les llama **capas ocultas.** La figura 6 muestra un esquema de una red neuronal artificial de tipo perceptrón multicapa, con una capa de entrada, una capa oculta y una capa de salida.

Las entradas de la red neuronal (En) son las características o atributos de nuestro problema. Las neuronas se representan con x_i^k, donde i corresponde al número de la neurona y k al número de la capa. Los pesos entre las neuronas se representan como $w_{i,j}^k$, que indica que es un peso entre las capas k y $k+1$ entre la neurona i de la capa k y la neurona j de la capa $k+1$. Los pesos entre las neuronas son los que contienen el aprendizaje de la red neuronal y se representan por una matriz de pesos Θ. La función

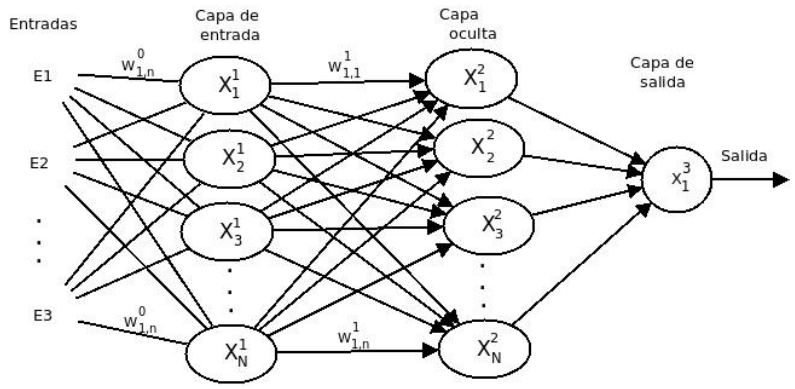

Figura 6: Ejemplo de una red neuronal artificial del tipo perceptrón multicapa.

de activación o de transferencia de cada neurona está dada por la fórmula:

$$a_i^k = g(\sum_{i=1}^{n} \Theta_i^k x_i^{k-1}) \tag{4}$$

En donde a_i^k representa la salida de una neurona, i es el número de la neurona, k es la capa en la red, Θ es una matriz de pesos que controla la función de mapeo de nuestra red, x_i^{k-1} son las neuronas de la capa anterior y $g()$ es la función de transferencia.

Aprendizaje

El aprendizaje de una red neuronal está dado por dos etapas: la primera etapa consiste en un algoritmo para propagar los datos a través de la red neuronal llamado **propagación hacia adelante**, la segunda etapa, llamada **propagación hacia atrás**, se encarga de corregir los pesos (aprendizaje) de la red neuronal. Estas dos etapas se repiten hasta minimizar el error entre la predicción de la red y los valores en nuestros datos de entrenamiento.

El primer algoritmo, llamado propagación hacia adelante, no modifica ningún peso de la red neuronal, se encarga de propagar la información producida por las neuronas entre cada capa hasta la capa de salida. Recordemos que la red neuronal cuenta con una

matriz de pesos, que regularmente es iniciada con todas sus entradas de manera aleatoria. De esta forma, la propagación hacia adelante se encarga de pasar la información de una capa de neuronas hacia la siguiente, sin alterar la matriz Θ. Esta información se calcula de la siguiente forma:

$$a_i^j = g(f(x))$$

$$h_\Theta(x) = g(\Theta_i^{j-1} a_i^{(j)})$$

Hay que notar que cada Θ_i^{j-1} es la matriz de pesos de la capa anterior. El tamaño de esta matriz esta dado por:

$$\Theta_{S_{j-1},(S_j)}$$

Donde S_{j-1} es el número de neuronas que tiene la capa anterior y S_j es el número de neuronas de la capa j. Cuando el algoritmo de propagación hacia adelante llega a la capa de salida se obtiene una predicción. Esta predicción dará información al algoritmo de propagación hacia atrás.

El segundo algoritmo, propagación hacia atrás (*Backpropagation*), encuentra el error en cada capa de nuestra red neuronal y cambia los valores que se tiene en la matriz de pesos. Minimizando el error entre la predicción y el *valor meta*, que proviene de los datos de entrenamiento y se considera como el valor correcto. Se puede ver a la matriz de pesos como el conocimiento de la red neuronal. Para encontrar el error y modificar los pesos de la matriz primero se obtiene el error cuadrado de la capa de salida:

$$\delta_i = (y_i - a_i^j)^2$$

La forma en la que se calcula el error en la capa de salida puede ser diferente, este cálculo depende de la función de activación que se va a usar, para este caso en particular se usa una función sigmoidea

El algoritmo de propagación hacia atrás trabaja de forma iterativa de la siguiente manera:

1. Dado un conjunto de entrenamiento $(x^{(1)}, y^{(1)}), ..., (x^{(m)}, y^{(m)})$

2. Asignamos $\Delta_{ij}^{(l)} = 0$ para toda l,i,j

3. Para $i = 1$ hasta m , con m el número total de ejemplos de entrenamiento.

   a) Asignamos $a^{(1)} = x^{(i)}$

   b) Hacemos propagación hacia adelante y con esto se calcula la función de activación para cada capa, el resultado lo guardamos en una $a^{(l)}$ para $l = 2, 3, ..., L$, con L el número total de capas que tiene la red neuronal.

   c) Calculamos el error entre el valor meta y el valor dado por la ultima capa de la red, $\delta^{(L)} = a^{(L)} - y^{(i)}$

   d) Calculamos el error para cada capa, $\delta^{(L-1)} = (\Theta^{L-1})^T \delta^L * (a^{L-1} * (1 - a^{L-1})), \delta^{(L-2)}, ..., \delta^{(2)}$

   e) Actualizamos las deltas, $\Delta_{ij}^{(l)} = \Delta_{ij}^{(l)} + a_j^{(l)} \delta_i^{(l+1)}$

4. Se minimiza el error acumulado en las deltas :

   - $D_{ij}^{(l)} = \frac{1}{m}\Delta_{ij}^{(l)} + \lambda \Theta_{ij}^{(l)}$ si $j \neq 0$
   - $D_{ij}^{(l)} = \frac{1}{m}\Delta_{ij}^{(l)}$ si j = 0

     Donde D es usada como un acumulador para sumar los valores a medida que avanzamos en el algoritmo, para que finalmente se calcula la derivada parcial de D y asi obtener que $\frac{\partial}{\partial \theta_{ij}^{(l)}} J(\theta) = D_{ij}^{(l)}$

Con estos dos algoritmos se busca reducir el error en la red neuronal en cada iteración. Dado un número de iteraciones se busca encontrar el menor mínimo local y lograr que el valor del error entre los conjunto de predicción y el conjunto de valores reales sea el mas pequeño posible Es importante resaltar que tanto la función para obtener el error en cada capa, así como la forma en la que se calcula el aumento de pesos en la red neuronal están relacionados directamente con el algoritmo de minimización que se esté

usando. En la sección 4.4 se describen los dos algoritmos que se usaron en esta tesis.

Las redes neuronales se han usado para resolver problemas de reconocimiento de números y letras, conducción automática de automóviles, diagnósticos médicos, reconocimiento de correo electrónico (spam), etc.

4.3. Funciones de activación

Una de las partes principales de una red neuronal artificial es la función de activación o de transferencia. Esta función hace el trabajo del potencial de activación en su contraparte biológica. Existen distintos tipos de funciones, cada una de ellas está enfocada para diferentes tipos de problemas a resolver, las que se mencionan a continuación se encuentran implementadas en la librería de aprendizaje automático con la que se ejecuta la red neuronal Tensorflow.

- **ReLU**: Unidad Lineal Rectificada, es una de las funciones de activación más populares, regularmente usada en las redes de convolución. Las redes de convolución son una variación de una perceptrón multicapa. La función ReLU es de la forma:

  $$f(x) = max(0,x)$$

  Otra variante es la función **softplus** que es de la forma:

  $$f(x) = log(exp(x)+1)$$

  Este tipo de funciones se le conocen como rectificadores y son usadas usualmente en visión por computadora y aprendizaje profundo.

- **Sigmoidea**: la función de activación sigmoidea está definida por:

  $$f(x) = \frac{1}{(1+e^{-x})}$$

La función hace que los erroes más pequeños convergen a 0 y los errores grandes convergen a 1. En los inicios de la redes neuronales fue una de las funciones más usadas.

- **Tanh**: la función tanh (tangente hiperbólica) nos dará valores entre [-1,1] pero centrada en 0. Por lo tanto, los valores más grandes convergen a 1 y los más pequeños a -1, mientras que la mayoría de los valores convergen a 0. La función Tanh esta dada por:

$$tanh(x) = \frac{sinh(x)}{cosh(x)}$$

La elección de la función de activación depende directamente del problema a resolver. Si se tiene un problema donde el comportamiento está definido en un intervalo se puede optar por la función sigmoidea; si lo que se quiere es la clasificación de múltiples objetos o hacer redes *deep learning* se puede optar por usar funciones de activaciones diferentes como es el caso de la función ReLU que es usada en compañia de otras funciones en la capa de salida como es la funcion softmax o incluso otras [38].

4.4. Algoritmo de minimización de error

Otra de las partes importantes de una red neuronal es el algoritmo de minimización del error. Este algoritmo se encarga de mejorar el conocimiento que se tiene en la matriz de pesos. El uso equivocado de un algoritmo de minimización puede provocar que el funcionamiento de una red neuronal se vea comprometido, dando como resultado una red que no trabaja de manera adecuada.

A continuación se mencionan dos algoritmos de minimización de error. Los dos son algoritmos de optimización que buscan encontrar los valores mínimos para los parámetros de una función, dado que estos parámetros son difíciles de calcular de manera analítica se hacen vía estos algoritmos.

Descenso por el gradiente(GD)

El descenso por el gradiente es un algoritmo iterativo usado para encontrar el mínimo de una función, siguiendo la dirección opuesta del gradiente. La función trabaja bien en funciones convexas pero el método es muy sensible a la elección del punto inicial que se tome, así que una mala elección del punto inicial hace al método susceptibles a errores.

La forma en que trabaja el algoritmo para cualquier caso, es:

1. Localizarse en un punto de la función y tratar de descender al punto mínimo usando la información de la primera derivada, entonces, tenemos que la primera derivada mide la rapidez con la que crece una función, al tener esta información podemos seguir la dirección contraria a la pendiente y con esto iremos bajando la pendiente hasta su punto mínimo.

2. Ya que sabemos hacia dónde tenemos que descender, hay que decidir cuánto descenderemos, para esto se utiliza un parámetro para cuantificar el tamaño del paso que se dará para descender, en el caso de la redes neuronal este parámetro se le conoce como taza de aprendizaje.

Para detener el proceso del descenso se puede hacer de dos manera: la primera es cuando la diferencia entre la nueva posición y la posición de donde se viene está por debajo de cierto umbral. La segunda opción, el proceso se detiene cuando se ha hecho un número definido de iteraciones.

La forma de trabajar del descenso por el gradiente en el caso específico de las redes neuronales es primero definir una función de costos, para después minimizar el error.

El algoritmo es el siguiente:

1. Primero se define la función de costos:

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} cost(\theta,(x^i,y^i))$$

$$cost(\theta,(x^i,y^i)) = \frac{1}{2}(h_\theta(x^i) - y^i)^2$$

2. Para j= 0...m , con m en número de capas que tiene la red neuronal

   a) Para i = 1 ... n, con n el número de neurona que tiene la capa j

      1) Calculamos el error en la capa j, $\theta_j = \theta_j - \alpha\frac{\partial}{\partial\theta_j} cost(\theta,(x^i,y^i))$

Donde h_θ es la función de activación y α es el índice de aprendizaje, por lo regular es de 0.01.

Descenso por el Gradiente Estocástico (SGD)

Al igual que GD, SGD se trata de un algoritmo iterativo que busca encontrar los valores mínimos de un función descrita por sumas de funciones diferenciables.

El algoritmo de SGD es igual que el de GD, la diferencia entre ellos es que SGD sólo se alimenta de una muestra de los datos de entrenamiento, es decir que se calcula el gradiente en funcion del conjunto de entrenamiento completo.

Los pesos se actualizan con mayor frecuencia, provocando que la convergencia se vuelva más rápida, aunque, el hecho de solo usar una muestra provoca que se tenga menor precisión en la actualización de los pesos. Sin embargo, esto no es del todo una desventaja dado que sirve para salir de los mínimos locales en los que puede caer el algoritmo.

Otra diferencia con GD es que la taza de aprendizaje se fija en la mayoría de los casos en 0.0001.

Principales diferencias entre Descenso por el gradiente(GD) y Descenso por el Gradiente Estocástico (SGD

Las principales diferencias entres estos dos algoritmos son:

1. GD se tiene que ejecutar en toda la muestra de los datos de entrenamiento para hacer la actualización en los parámetros a minimizar mientras que SGD utiliza una muestra para actualizar los parámetros desde la primera iteración.

2. SGD converge de manera más rápida que GD; por lo tanto, los valores obtenidos en la mayoría de los casos son los óptimos y se queda oscilando en ellos [31].

3. SGD mantiene la taza de aprendizaje por parámetro, lo que le permite mejorar el rendimiento en problemas con escasos datos o con ruido.

Metodología

El sistema para el pronóstico de ozono cuenta con 22 redes neuronales, una para cada estación del sistema de monitoreo atmosférico de la Ciudad de México. Cada red neuronal esta entrenada con la información de los contaminantes obtenida de las estaciones meteorológicas así como el pronóstico meteorológico obtenido del CCA. Para la programación del sistema se hace uso de python junto con la librería de aprendizaje automático Tensorflow.

Para el desarrollo de la red neuronal se define el ambiente de desarrollo (tabla 3), en el cual se específica en primer lugar la base de datos en donde se guarda la información, el lenguaje de programación con el que se trabajó y la librería de aprendizaje automático en la cual está implementada la red neuronal. Además de identificar las estaciones con las que se trabaja, los datos que se le dará a la red neuronal, el pre-procesamiento que se le hace a los datos y por último las pruebas de validación.

Concepto	**Herramienta**	**Versión**
Lenguaje de programación	Python	3.5
Diseñador de DB	DbScheme	3.14
Manejador de base de datos	Postgresql	1.22.1
Libería de aprendizaje automático	Tensorflow	1.0

Tabla 3: Ambiente de desarrollo para la red neuronal artificial.

Los datos con los que está entrenada la red neuronal se obtienen del Sistema de Monitoreo Atmosférico de la Ciudad de México (http://www.aire.cdmx.gob.mx). El sistema cuenta con información de 67 estaciones pero sólo se usa información de las 22 estaciones atmosféricas que aún se encuentran en funcionamiento (Figura 7). Cada estación atmosférica recolecta información de 7 contaminantes (tabla 4), un dato por cada hora del día.

Siglas	**Nombre**
PM_{10}	Partículas menores a 10 micrómetros
$PM_{2,5}$	Partículas menores a 2.5 micrómetros
NO_X	Óxidos de nitrógeno
NO_2	Dióxido de nitrógeno
CO	Monóxido de carbono
O_3	Ozono
SO_2	Dióxido de azufre

Tabla 4: Contaminantes con los que se entrenan las redes neuronales.

Además de los contaminantes, se obtienen datos de la predicción a 24 horas de datos meteorológicos (tabla 5), dicha predicción proviene del pronóstico realizado por el grupo IOA del CCA.

Las 22 estaciones seleccionadas se muestran en la tabla 6 y en la figura 7 se muestra la posición geográfica de las estaciones, dentro de la Ciudad de México así como en el área metropolitana.

Ya definidas las estaciones a usar, se concretó el diseño para la base de datos, dichos datos se almacenan en una base de datos relacional usando SQL, tanto los script de petición de datos, así como la creación y llenado de la base de datos se encuentra disponible en github [37].

Siglas	Nombre
U10	Vientos de oeste a este a 10 mts sobre la superficie
V10	Vientos de sur a norte a 10 mts sobre la superficie
RAINC	Precipitación acumulada en 'mm' de todos los procesos convectivos
T2	Temperatura a 2 mts sobre la superficie
TH2	Temperatura potencial a 2 mts
RAINNC	Precipitación acumulada de las fases no conectivas
PBLH	Altura de la capa de mezcla
SWDOWN	Flujo de la radiación de onda corta
GLW	Flujo de la radiación de onda larga

Tabla 5: Datos meteorológicos con los que se entrenan las redes neuronales.

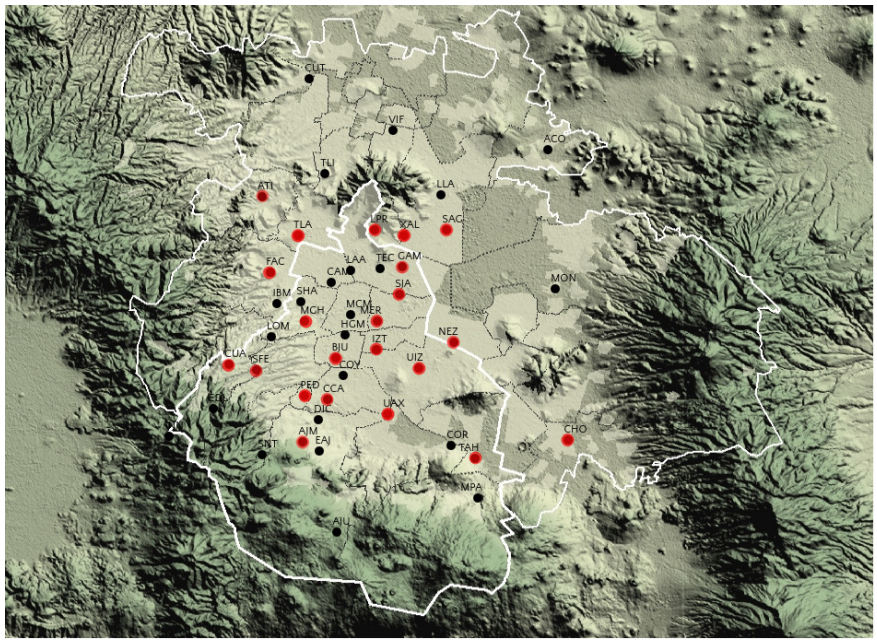

Figura 7: Mapa con las 22 estaciones que se tomaron en cuenta para entrenar la red neuronal (rojos) y el resto de las estaciones de la RAMA (negros).

5. Pre-procesamiento

Antes del diseño y construcción de la red neuronal hay que dar un tratamiento a los datos. Lo primero que se tiene que asegurar es la integridad de éstos, es decir, que no

Siglas	Nombre	Siglas	Nombre
AJM	Ajusco Medio	MGH	Miguel Hidalgo
ATI	Atizapan	NEZ	Nezahualcóyotl
BJU	Benito Juárez	PED	Pedregal
CAM	Camarones	SAG	San Agustín
CCA	Centro de Ciencias de la Atmósfera	SFE	Santa fe
CHO	Chalco	SJA	San Juan Aragón
CUA	Cuajimalpa	TAH	Tlahuac
FAC	FES Acatlán	TLA	Tlalnepantla
IZT	Iztacalco	UAX	UAM Xochimilco
LPR	La Presa	UIZ	UAM Iztapalapa
MER	Merced	XAL	Xalostoc

Tabla 6: Estaciones atmosféricas seleccionadas de la pagina http://www.aire.cdmx.gob.mx/default.php dando un total de 22 estaciones.

se tengan datos nulos o que sean no numéricos. En este caso, se determinó que si se encuentran datos nulos o no numéricos, se les asigna el valor de -1. Se decide usar el valor de -1 y no de 0, ya que el valor de 0 representa ausencia del contaminante. Es importante verificar que la mayoría de los datos estén completos, es decir, que en cada ejemplo de entrenamiento se tenga información de la mayoría de los contaminantes.

Para saber qué estaciones tienen datos faltantes en el rango de tiempo de interés se hicieron gráficas como la de la figura 8. Esta gráfica muestra, con color negro, los huecos en el tiempo dónde no hay contaminantes. En el eje X se tienen los contaminantes que se usan para entrenar la red neuronal. Una vez identificados los datos faltantes se probaron tres posibles alternativas para el manejo de los datos.

1. Dejar tal cual los datos que se tienen, es decir, sólo se asegura la integridad de

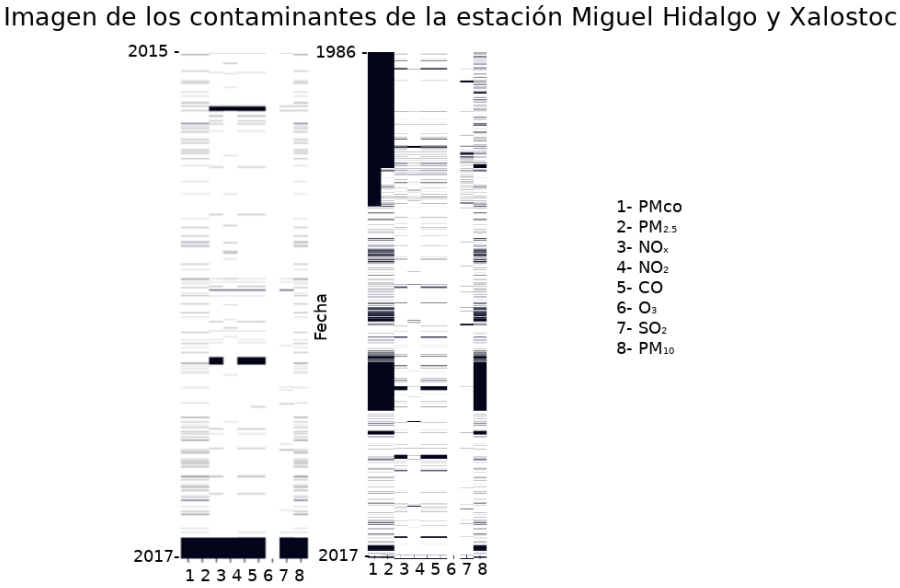

Figura 8: Imagen de los datos de contaminantes de la estación Miguel Hidalgo y Xalostoc, las lineas de color negro representan datos nulos.

los datos.

2. Quitar los atributos (columnas) de aquellos contaminantes de los cuales no se tiene ningún dato.

3. Quitar cada ejemplo de entrenamiento (renglón) dónde falte información de algún contaminante.

Tener estas opciones permite poder analizar la tolerancia de la red neuronal a la falta de información. Los resultados de cada alternativa se presentan en la sección 13.

Los datos meteorológicos se obtienen a través de archivos NetCDF. Estos archivos contienen información meteorológica de todo el país, por lo que es necesario cortarlos a la región de la CDMX y área metropolitana. Cada archivo NetCDF contiene el pronóstico horario para los próximos 5 días de cada variable meteorológica, Esta in-

formación es en tres dimensiones (latitud, longitud y tiempo) y se puede ver como un cubo de datos para cada variable. Cada cubo tiene 120 *niveles*, cada nivel representa una hora del día. A partir de esta información se debe decidir qué datos se incorporan a la red neuronal. En este trabajo se propone agregar la información meteorológica de las próximas 24 horas a la hora *actual* para hacer el pronóstico. Por ejemplo, si queremos pronosticar cual será la concentración de Ozono para la estación del CCA en 24 horas, lo que se ingresa a la red neuronal son: los datos de contaminación de la estación del CCA de la hora actual y los datos meteorológicos del pronóstico de las próximas 24 hrs. Para incorporar la información meteorológica a la red neuronal se propusieron las siguientes 4 formas.

1. No agregar los datos de meteorología.

2. Obtener el promedio de la región por las próximas 24 horas. En esta forma se agregan 24 atributos a la red neuronal por cada variable meteorológica, un total de 216.

3. Obtener el promedio de cuatro subcuadrantes de la región por las próximas 24 horas. En esta forma se agregan 96 $(24 * 4)$ atributos por cada variable meteorológica, un total de 864 $(24 * 4 * 9)$.

4. Obtener el promedio de 16 subcuadrantes de la región por las próximas 24 horas. En esta forma se agregan 384 $(24 * 16)$ atributos por cada variable meteorológica, un total de 3456 $(24 * 4 * 9)$.

Al igual que en el caso de los contaminantes, se prueba cada una de las 3 alternativas con el fin de ver el comportamiento de la red neuronal.

Ya procesados los datos de los contaminantes y los meteorológicos, sigue unirlos en una sola BD. Aquí surge uno de los primeros problemas a resolver. Dado que la

información que se tiene de los datos meteorológico son del año 2010 al año 2017 y la de los contaminantes para algunas estaciones va desde el año 1986, esto generó un problema de desigualdad de datos. Para resolver el problema se decidió reducir la información de los contaminantes a partir el año 2010.

6. Bootstrap

La poca cantidad de datos en algunas estaciones afecta de manera directa el desempeño en la predicción de O_3, para obtener más datos se usa la técnica de bootstrap. Bootstrap es una técnica de re-muestreo usada principalmente en problemas que se enfocan en construir los intervalos de confianza o para aproximar el sesgo o la varianza de un análisis estadístico. Esta técnica se usa para obtener datos adicionales a partir de los datos iniciales.

La técnica de bootstrap, propuesta por Bradley Efron en 1979 [4] se le conoce como una técnica de re-muestreo, toma una cantidad de datos de manera repetida de los datos iniciales, hace inferencias estadísticas y repite estos pasos de manera iterativa, generando más datos de entrenamiento. La forma en que se aplicó esta técnica en nuestro problema fue la siguiente:

1. Sacar la media de nuestro contaminante O_3 para cada estación en el rango de nuestros datos

2. Sacar la desviación estándar.

3. Duplicar aquellos ejemplos cuyo valor supere al promedio más una desviación estándar.

Aunque es una de las técnicas más usada en el re-muestreo existen otras como las de Jacknife [10], pruebas de aleatorización y permutación o prueba de validación cruzada.

Sin embargo, se decidió usar bootstrap con el fin de reducir el error de la predicción cuando los valores de contaminantes son altos, es decir los picos de los contaminantes, ya que se tiene un número pequeño de ejemplos de esta situación.

7. Normalización

El funcionamiento de la red neuronal puede verse comprometido cuando tenemos atributos cuyo rango de valores es muy distinto. Para que esto no ocurra se recomienda normalizar los datos. La normalización de datos consiste en poner los valores de todas las variables en el rango de 0 a 1. Para normalizar los datos se usa la siguiente fórmula:

$$x_n = \frac{x_i - V_s}{V_s - V_i}$$

Donde x_i es el valor original de la variable para el ejemplo i. V_s es el valor superior que puede tomar la variable y V_i es el valor inferior. Para la normalización de los datos se uso la biblioteca sklearn (http://scikit-learn.org/stable/), en especifico la clase de preprocessing. Además de normalizar los datos, se guardaron los valores máximos y mínimos de cada variable, para "desnormalizar" el pronóstico generado por la red neuronal.

8. División de datos

Ya que se tiene asegurada la integridad de los datos y se han normalizado, el siguiente paso es dividir los datos en tres conjunto: el conjunto de entrenamiento, el de validación y el de prueba.

La razón de dividir nuestros datos en estos tres conjuntos es para poder cuantificar la eficacia y el correcto funcionamiento de la red neuronal.

La división de los datos quedó conformada de la siguiente manera:

1. El conjunto de entrenamiento tiene los datos del año 2010 al año 2015.

2. El conjunto de validación tiene los datos del año 2016.

3. El conjunto de prueba tiene los datos del año 2017

Para probar la eficacia de las diferentes configuraciones de datos y de las diferentes arquitecturas para la red neuronal, cada red se entrenó con el conjunto de entrenamiento y para evaluar dichas configuraciones se usó el conjunto de validación, este conjunto de pruebas ayudó a definir cuál es la mejor configuración de datos y de arquitectura de la red neuronal.

Ya definida la arquitectura de la red y la configuración de datos que se van a usar, lo siguiente es verificar la configuración con el conjunto de prueba, esta es usada para verificar que no existe un sobreajuste en el entrenamiento de la red neuronal.

9. Arquitectura de la red neuronal

Para la red neuronal se optó por usar una red de arquitectura Perceptrón Multicapas, este tipo de red tiene una topología de conexión hacia adelante (Figura 6). La arquitectura está diseñada por una capa de entrada, n-capas ocultas y una capa de salida.

Nuestra capa de entrada está formada por n nodos, donde n depende del número de contaminantes y de los datos meteorológicos con lo que se entrena la red neuronal. El diseño de la red neuronal sólo cuenta con una capa oculta que tiene n^2 nodos y la capa de salida tiene una sola neurona, cuyo valor de salida corresponde a la cantidad de ozono esperada 24 horas después.

Otros aspectos que se tienen que definir en el diseño de la red neuronal son: el índice de aprendizaje, el algoritmo de minimización del error, la función de activación y el número de iteraciones con los que se entrena la red neuronal, todos estos parámetros

son fijados con el conjunto de validación.

El índice de aprendizaje se fijó en un valor que depende del algoritmo de minimización. Se usaron dos algoritmos de minimización, GD y SGD, la función de activación seleccionada fue una función sigmoidea.

Se hicieron múltiples pruebas para buscar la mejor configuración del sistema de redes neuronales. La primera prueba que se hizo fue respecto a la forma de leer los datos, con el fin de minimizar el tiempo de lectura y procesamiento. Las siguientes pruebas se pueden agrupar en pruebas que investigan cuál es la configuración óptima para la red neuronal. Y las últimas pruebas se enfocan en encontrar cual son los mejores datos a suministrar a la red neuronal para su entrenamiento.

10. Tiempo en la carga de los datos

Cuando se diseña una red neuronal se requiere de muchos intentos que son costosos computacionalmente. Si nuestro sistema se tarda varias horas en leer los datos de entrenamiento, entonces estamos limitados al número de pruebas que podemos realizar. Para resolver o minimizar el impacto de dicho problema, se busca la mejor manera de almacenar y leer datos. Los datos de los contaminantes están almacenados en una base de datos de PostreSQL. Se analizó leer directamente los datos de la base de datos contra generar un archivo .csv (archivo con valores separados por comas) con la información de entrenamiento y leer directamente el archivo csv. Los datos meteorológicos están almacenados en archivos netCDF. Para estos datos se requiere un postprocesamiento y los resultados de procesamiento se guardan archivos csv.

La prueba consiste en estudiar el tiempo de obtención de datos vía petición a la base

de datos contra cargar los archivos .csv que contienen los datos de los contaminantes. Los tiempos de lectura se obtienen de forma incremental, primero sólo se leen los datos de una estación, después de dos estaciones, luego de tres y así sucesivamente.

11. Número de iteraciones

En el funcionamiento de una red neuronal el número de iteraciones de entrenamiento influye en el tiempo de ejecución y en la predicción de la red neuronal. Si se tiene un número de iteraciones pequeño puede provocar que la red neuronal no alcance a aprender los datos de entrenamiento. Mientras que un número grande de iteraciones puede aumentar el tiempo de cómputo y provocar un sobreajuste en el entrenamiento de la red neuronal.

Para encontrar el número de iteraciones para cada configuración de datos y parámetros con los que se entrena cada red neuronal, haremos uso de una función que esta implementa dentro de tensorflow. Dicha función hace un monitoreo del entrenamiento y detiene dicho entrenamiento cuando el error se mantiene estable, de esta manera se encuentra el mejor número de iteraciones para entrenar la red neuronal y evitar un sobreajuste en el entrenamiento de la red neuronal.

Para esta prueba se usa la estación XAL (Xalostoc) con las 4 configuraciones: *GCC*, *CM*, *CC* y *LCB*; dichas configuraciones están descritas en la sección 12. Tomaremos estas 4 configuración como las principales, a partir de estas pruebas las demás configuraciones son entrenadas con el número de iteraciones que resulten de esta prueba. La prueba se hace con el conjunto de entrenamiento y el validacion, de esta manera podemos prevenir un sobreajuste en el entrenamiento o una falta de entrenamiento de las redes neuronal.

12. Entrenamiento de la red neuronal

Ya que se ha fijado la forma en que se obtienen los datos, la topología de la red neuronal , el siguiente paso es probar la configuración de datos con las que es entrenada la red neuronal, así como parte del algoritmo de entrenamiento de la red neuronal que es la función de minimización.

Se tienen un total de 14 configuraciones que se resumen en la tabla 7. De cada configuración se obtiene una gráfica de predicción de cada estación y el índice de correlación y Utheils (estas métricas de describen en la sección 13) por cada estación así como el promedio de todas las estaciones. Para estas pruebas las redes se entrenan con el conjunto de entrenamiento y se prueba con el conjunto de validación

Las opciones de la columna **Datos Contaminantes** son los siguientes:

- **Completos**. Datos originales.
- **Limpios**. Se eliminan los datos nulos.
- **Completos Bootstrap**. Datos originales con remuestreo usando Bootstrap.
- **Limpios Bootstrap**. Datos sin valores nulos con remuestreo usando Bootstrap.

Las opciones de la columna **Datos Meteorológicos** son los siguientes:

1. **Promedio**. Se toma la matriz que contiene cada dato meteorológico y se saca el promedio de toda la matriz. De esta manera tendremos un promedio del contaminante para cada hora del pronóstico, con lo que se obtienen 24 datos por contaminante.
2. **Cuadrantes.**. Se divide la matriz de información en cuadrantes y para cada cuadrante se obtiene el promedio. En este casos se tiene que por cada hora del

Siglas	**Datos Contaminantes**	**Datos Meteorológicos**	**Func. minimización de error**
CC	Completos	Cuadrantes	Sthocastic Gradient
CC16	Completos	Cuadrantes 16	Sthocastic Gradient
CP	Completos	Promedio	Sthocastic Gradient
CM	Completos	sin datos	Sthocastic Gradient
GCC	Completos	sin datos	Gradient Descent
CCB	Completos(Bootstrap)	Cuadrantes(Bootstrap)	Sthocastic Gradient
CPB	Completos(Bootstrap)	Promedio(Bootstrap)	Sthocastic Gradient
LC	Limpios	Cuadrantes	Sthocastic Gradient
LC16	Limpios	Cuadrantes16	Sthocastic Gradient
LP	Limpios	Promedio	Sthocastic Gradient
LM	Limpios	sin datos	Sthocastic Gradient
GLC	Limpios	sin datos	Gradient Descent
LCB	Limpios(Bootstrap)	Cuadrantes(Bootstrap)	Sthocastic Gradient
LPB	Limpios(Bootstrap)	Promedio(Bootstrap)	Sthocastic Gradient

Tabla 7: Configuración de datos y arquitectura para probar la capacidad de predicción

pronóstico obtendremos 4 datos y por cada variable se tiene un total de 24*4 datos.

3. **Cuadrantes16**. Se divide la matriz en 16 subregiones y para cada región se obtiene el promedio.

La última columna indica el algoritmo de minimización de error. La mayoría de las pruebas se hicieron con SGD debido a que rápidamente se observó mejor desempeño que GD.

Una columna omitida fue la de la funcion de activacion, para todas las configuraciones se usa la funcion sigmoidea, la razon de porque se usa esta funcion de activacion se

debe al comportamiento en la emision de ozono. En la figura 9 se muestra la emisión de ozono para los cinco primeros días del mes de junio para los años 2010 al 2017, dicha figura nos deja ver que el comportamiento de emisiones de ozono, en las que podemos notar que son regulares en su forma pero cambian en cuanto a su magnitud. A pesar de que la funcion sigmoidea no es identica en cuanto a su forma, es la funcion que mas se acerca al comportamiento, teniendo un valor mimino cercano a 0 y un valor maximo que en el caso de la emision de ozono se puede tomar con el mayor valor historico de los 7 años de informacion.

Otra información que se le da a la red neuronal son: el día la semana, el año, el mes, el día, si es día feriado o no y si hubo o no contingencia.

Se obtiene la mejor configuración de datos y la mejor arquitectura para el funcionamiento de las redes neuronales, Ya que se tiene la mejor configuración de estas 14, la red se entrena con el conjunto de entrenamiento y el de validación, y se prueba dicho entrenamiento con el conjunto de prueba. Se calcula el índice de correlación y Utheils para cada estación.

El resultado de cada prueba descrita anteriormente se encuentran en la sección de Resultados 13.

13. Métricas

Para medir el desempeño de un pronóstico es importante contar con un índice de precisión, la precisión se define como la diferencia entre un valor pronosticado y un valor real, pero, ¿para qué necesitamos evaluar la precisión de un pronóstico? En un principio, medir el desempeño del pronóstico es importante para saber que los métodos y características con los que trabaja el pronóstico es el adecuado. Además, calcular la

precisión nos ayudará en la búsqueda de partes del pronóstico que no trabajen de manera óptima y con esto tomar decisiones para mejorar el desempeño del mismo.

Para obtener el valor de la precisión hay que obtener el error. ¿De dónde sale el error en un pronóstico?, hay dos fuente principales: la primera, se conoce como **error sistemático**, este es causa de un error constante, como puede ser tomar variables equivocadas o que están poco relacionadas. La segunda fuente, se le conoce como **error aleatorio**, este tipo de error no tiene explicación, es decir, ocurre de manera imprevista y no se le puede dar una explicación concreta.

El cálculo de la precisión está basado en la siguiente fórmula:

$$e = (V_o - V_p)$$

Donde "e" es el valor del error, V_o es el valor observado y V_p es el valor pronosticado. Otra forma de obtener el error pero esta vez en forma de porcentaje es con la siguiente fórmula:

$$ea = \frac{|V_o - V_p|}{V_o} * 100$$

Existe una gran cantidad de métricas para medir el desempeño de un pronóstico lo cual hace que la elección de la más adecuada se vuelva complicada. Es importante mencionar que para elegir una métrica se tiene que tomar en cuenta qué tanto se quiere penalizar a las predicciones con errores grandes. A continuación se mencionan algunas métricas usadas para medir el desempeño de los pronósticos. Para el presente trabajo se usa el **índice de correlación**, así como **U de Theils**. El uso del índice de correlación es una de las métricas más usadas para medir la precisión de los pronósticos, mientras que U de Theils se eligió para comparar las diferentes formas en que se penalizan los errores.

Métricas principales

Índice de correlación

Cuando el índice de correlación es 1 nos indica que hay una relación perfecta uno a uno y si es -1 hay un relación inversa entre ellas.

$$\rho_{xy} = \frac{Cov_{xy}}{\sigma_x \sigma_{[y]}}$$

Donde Cov_{xy} es la covarianza entre x e y, σ_x es la desviación típica x y σ_y es la desviación típica de y.

U de Theil

Esta métrica se basa en la diferencia cuadrática entre la tasa de crecimiento del valor observado y de la predicción. En el coeficiente de desigualdad de U cuando el valor es cercano a 0 se dice que la predicción es perfecta, hecho difícil de lograr,

$$U_t = \sqrt{\frac{\sum_{t=1}^{n} e_t^2}{\sum_{t=1}^{n} (V_{ot} - V_{ot-1})^2}}$$

Otras métricas

MPE (Mean Percentage Error)

Métrica usada para ver si el error tiene un sesgo o si el pronóstico está sobrestimado o subestimado:

$$MPE = \frac{\sum_{t=1}^{n} \frac{(V_{ot} - V_{pt})}{V_{ot}}}{n}$$

Donde V_{ot} es el valor observado y V_{pt} es el valor calculado o de predicción

MAPE (Mean Absolute Percentage Error)

MAPE es de las medidas más utilizadas para saber la precisión de los pronósticos. Esta métrica no considera el signo del error solo toma la magnitud.

$$MAPE = \frac{\sum_{t=1}^{n} \frac{|V_{ot} - V_{pt}|}{V_{ot}}}{n}$$

Index of agreement

El denominador Index of agreement se conoce como el error potencial, cuando el valor del índice es cercano a 1 nos indica que hay un buen acuerdo entre el pronóstico y el valor observado

$$IA = 1 - \frac{||(V_{pt} - V_{ot})^2||}{||(|V_{pt} - ||V_{pt}||| + |V_{ot} - ||V_{ot}||)^2||}$$

NMSE (Normalized Mean Square Error)

En el estimador NMSE se suman las desviaciones en lugar de las diferencias, por lo tanto se muestra mejor la diferencia entre la predicción y el valor observado. Cuando el valor obtenido es bajo, se tiene una buena predicción, mientras que cuando el valor es alto se tiene que el modelo está equivocado.

$$NMSE = \frac{1}{n} \sum_{t=1} \frac{(P_t - M_t)^2}{PM}$$

$$P = \frac{1}{n} \sum_{t=1} V_{pt}$$

$$V = \frac{1}{n} \sum_{t=1} V_{ot}$$

Donde tenemos que V_{ot} es el valor observado mientras que V_{pt} es el valor calculado o de predicción.

RMSE (Root Mean Square Error)

Cuando la desviación cuadrática media de la raíz nos da valores pequeños nos indica que hay una buena predicción para los valores observados.

$$\sqrt{||(V_{pt} - V_{ot})^2||}$$

Índice de Concordancia

El índice de concordancia o también conocido como el coeficiente de kappa mide la concordancia entre dos clasificadores. Cuando el índice es 1 hay un completo acuerdo entre los dos clasificadores, cuando es diferente no hay acuerdo entre los dos clasificadores.

$$k = \frac{V_{ot} - V_{pt}}{1 - V_{pt}}$$

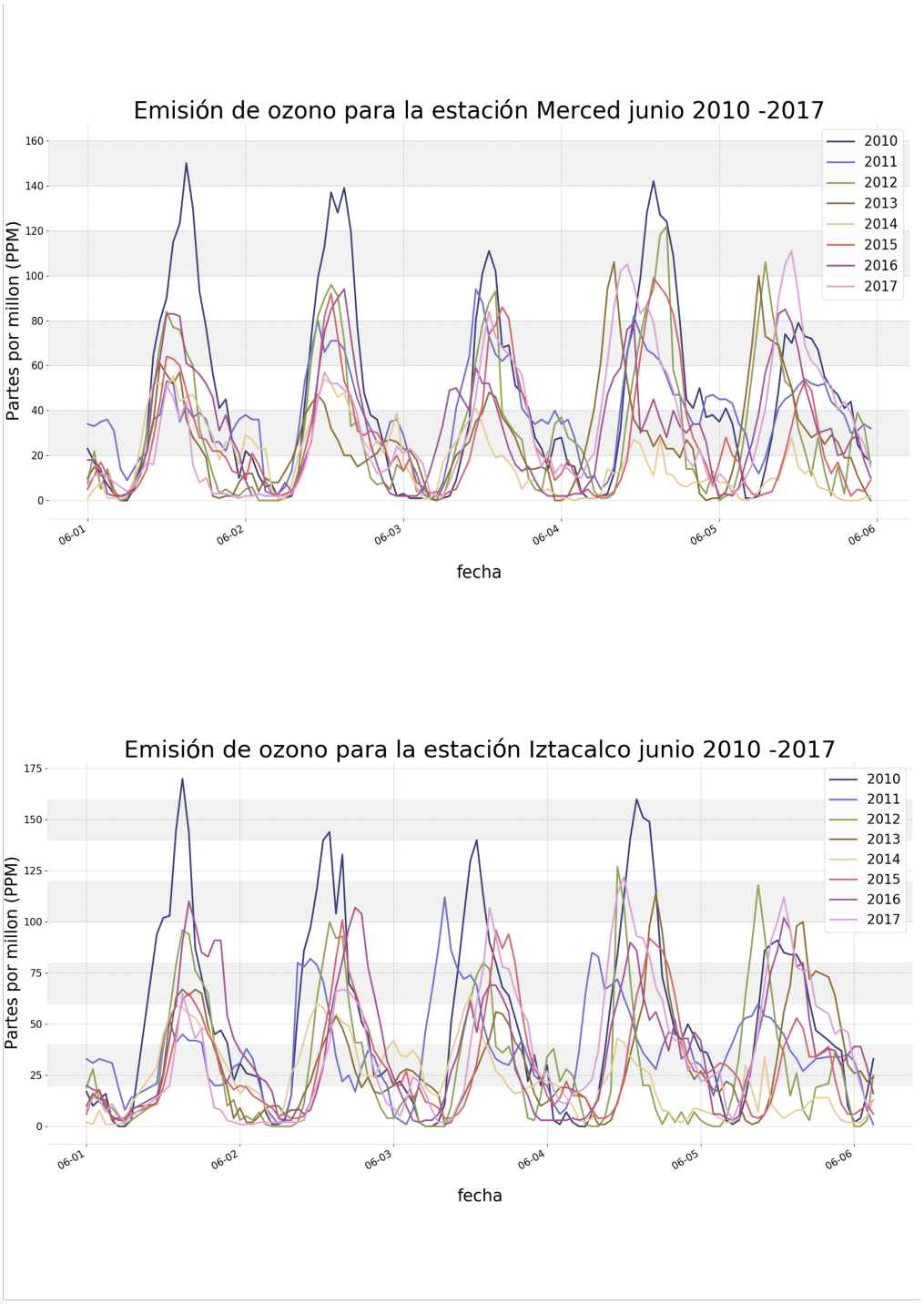

Figura 9: Gráficas de la emisión de ozono de las estaciones Merced y Iztacalco para los cinco primeros días de junio de los años 2010 al 2017.

Resultados

En esta sección se mencionan los resultados de las diferentes pruebas, empezando con los resultados obtenidos para el número de iteraciones con las que se entrenan las redes neuronales. Posteriormente se tienen los resultados en los tiempos de lectura de datos. Por último los de mayor relevancia, los resultado de entrenar con diferentes configuraciones de datos la red neuronal.

14. Pruebas para el funcionamiento de la red neuronal

14.1. Número de iteraciones

Las figuras 10 y 11 muestran el error contra el número de iteraciones, donde se observa el número óptimo para cada una de las cuatro configuraciones con las que se hizo la prueba.

Para la primer configuración *GCC* la cual tiene como algoritmo de minimización descenso por el gradiente, notamos que es la configuración con mayor error tanto en el entrenamiento como en la validación. Un caso particular es lo que sucede con la configuracion *CC*, en la prueba hecha con el conjunto de entrenamiento, se observa que el comportamiento del error se parece al de la configuración *LCB*, sin embargo en la prueba de validación vemos un claro sobreajuste en el entrenamiento de la red neuronal.

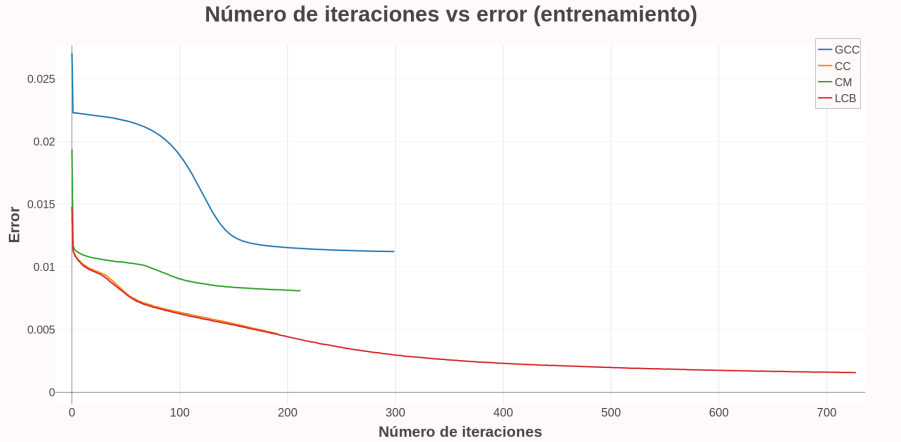

Figura 10: Gráfica del error vs el número de iteraciones de las 4 configuraciones con el conjunto de entrenamiento

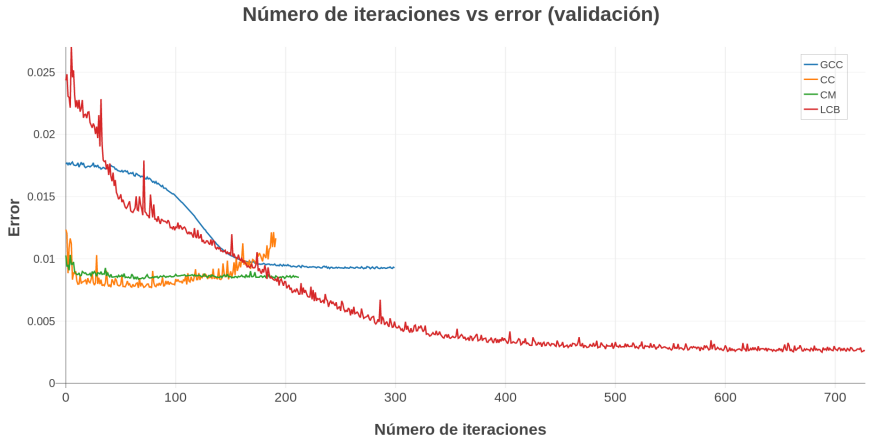

Figura 11: Gráfica del error vs el número de iteraciones de las 4 configuraciones con el conjunto de validación

Como resultado de esta prueba se desprende una prueba extra, esta se basa en la configuración CC, la cual tiene un claro sobreajuste. Para esta prueba se modificó el número de neuronas que contiene las capas. Se hicieron dos experimentos, para el primer experimento se aumentó el número de neuronas mientras que para el segundo se disminuye el número de neuronas, al igual que en la prueba anterior se toma el error en

con el conjunto de entrenamiento y con el conjunto de validación.

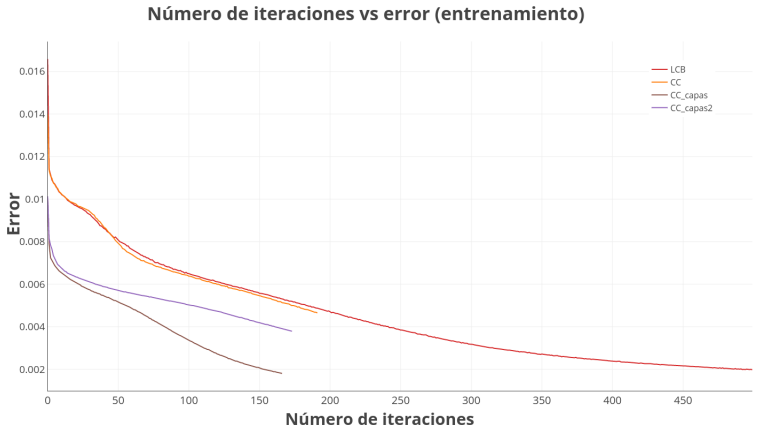

Figura 12: Gráfica del error vs el número de iteraciones, para la configuracion CC, con el conjunto de entrenamiento

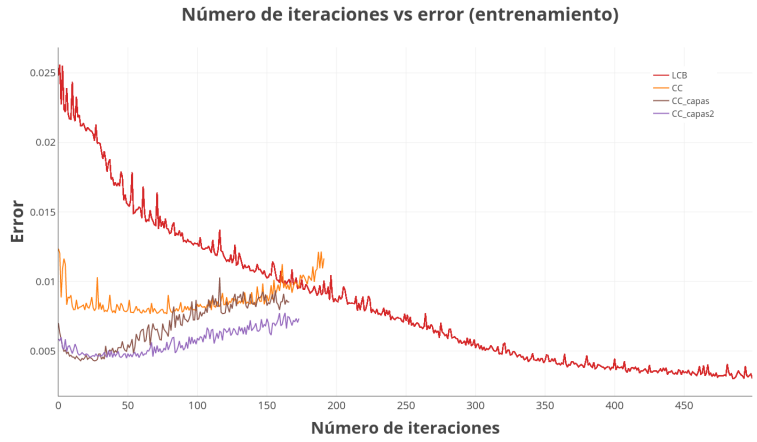

Figura 13: Gráfica del error vs el número de iteraciones, para la configuracion CC, con el conjunto de validacion

Como resultado de modificar el número de neuronas, obtenemos que en los dos experimentos hechos se mantiene el sobreajuste (figura 12 y 13). Con este resultado, fijamos el número de neuronas y el número de iteraciones con los valores encontrados en la configuración CC original.

En la tabla 8 se muestra el número de iteraciones que se fijó para cada configuración así como el error que se obtuvo tanto en el entrenamiento y en la validación.

Configuración	Número de iteraciones	Error entrenamiento	Error validación
GCC	250	0.0112	0.0092
CM	200	0.0081	0.0085
CC	100	0.0046	0.0116
LCB	700	0.0015	0.0025

Tabla 8: Número de iteraciones y error para cada configuración

14.2. Lectura de datos

La figura 14 muestra el tiempo que el sistema tarda en obtener los datos desde la base de datos contra cargar los datos desde archivos .csv previamente generados. El tiempo incluye la unión con los datos meteorológicos que fueron antes procesados de archivos netCDF. Los resultados se muestran de manera incremental con el número de estaciones que se procesan.

La diferencia de tiempo se hace notable desde que se cargan dos estaciones. El tiempo de carga de los archivos .csv es cerca de 100 veces menor que el de la base de datos cuando se tiene que cargar la información de todas las estaciones.

Obtener los datos de los contaminantes desde archivos .csv es la mejor forma de disminuir el tiempo de carga de datos, este hecho permite agilizar las pruebas con las diferente configuraciones de datos que se muestran a continuación.

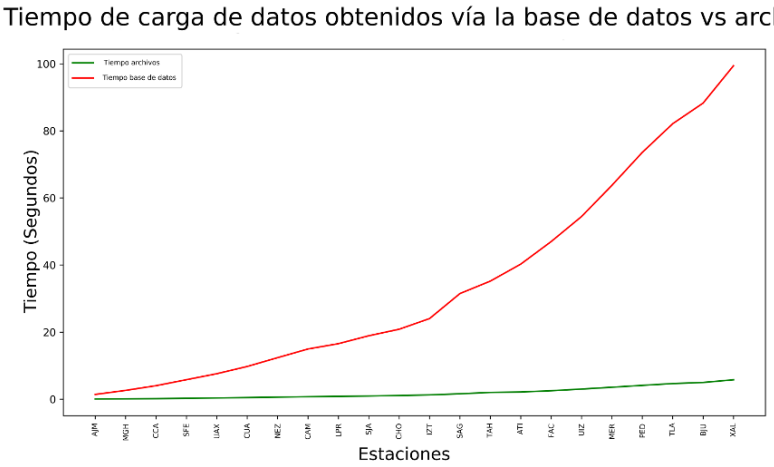

Figura 14: Tiempo que tardan en cargarse los datos de contaminantes desde la base de datos contra leer archivos csv creados con anterioridad.

15. Pruebas de entrenamiento con diferentes configuraciones de datos.

Ya fijo el número de iteracionesy la forma en que se cargan los datos a la red neuronal, se prueba la arquitectura de la misma.

15.1. Resultados GCC (Datos completos, sin datos meteorológicos, GD como función de minimización)

La primera prueba de entrenamiento se realizó con datos de los contaminantes (sin meteorología) y Gradient Descent como función de minimización. El resultado de la primer prueba se muestra en la figura 15.

El resultado de la primer prueba muestra un índice de correlación muy bajo, las mejores estaciones tienen un índice de correlación de 0.5 mientras que podemos ver en la gráficas de predicción del Ajusco medio (figura 16) que la predicción hecha por la

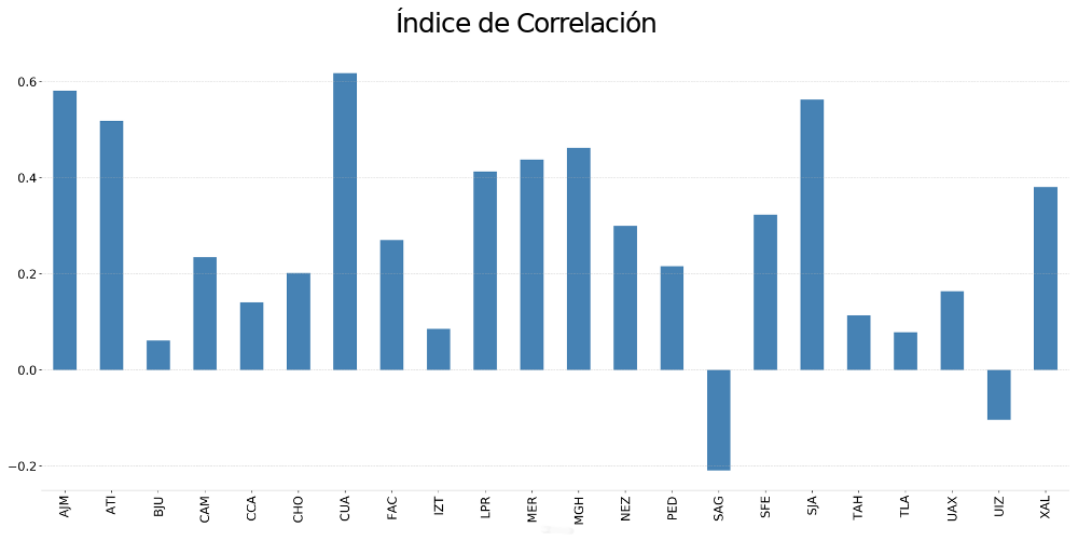

Figura 15: Indice de correlación para la configuración GCC.

red neuronal está muy lejos de los valores observados. Sin embargo, en algunos puntos se observa que la red neuronal es capaz de seguir el comportamiento de las emisiones.

La correlación que logra la red neuronal para esta configuraciones es de 0.211 en promedio mientras que el índice de Utheils es de 0.496. Además de la gráfica de correlación, se muestra la predicción hecha por la red neuronal para la estación del Ajusco Medio, Benito Juárez y Xalostoc en la figura 16 para el año 2016.

Uno de las razones del mal desempeño de las redes neuronales se le puede atribuir a la configuración de la red neuronal, es decir, que algunos de los componentes no sea el óptimo para el tipo de problema que se quiera solucionar. Otra posible razón es la arquitectura de la red neuronal, hecho que se puede observar en la grafica 10 y 11.

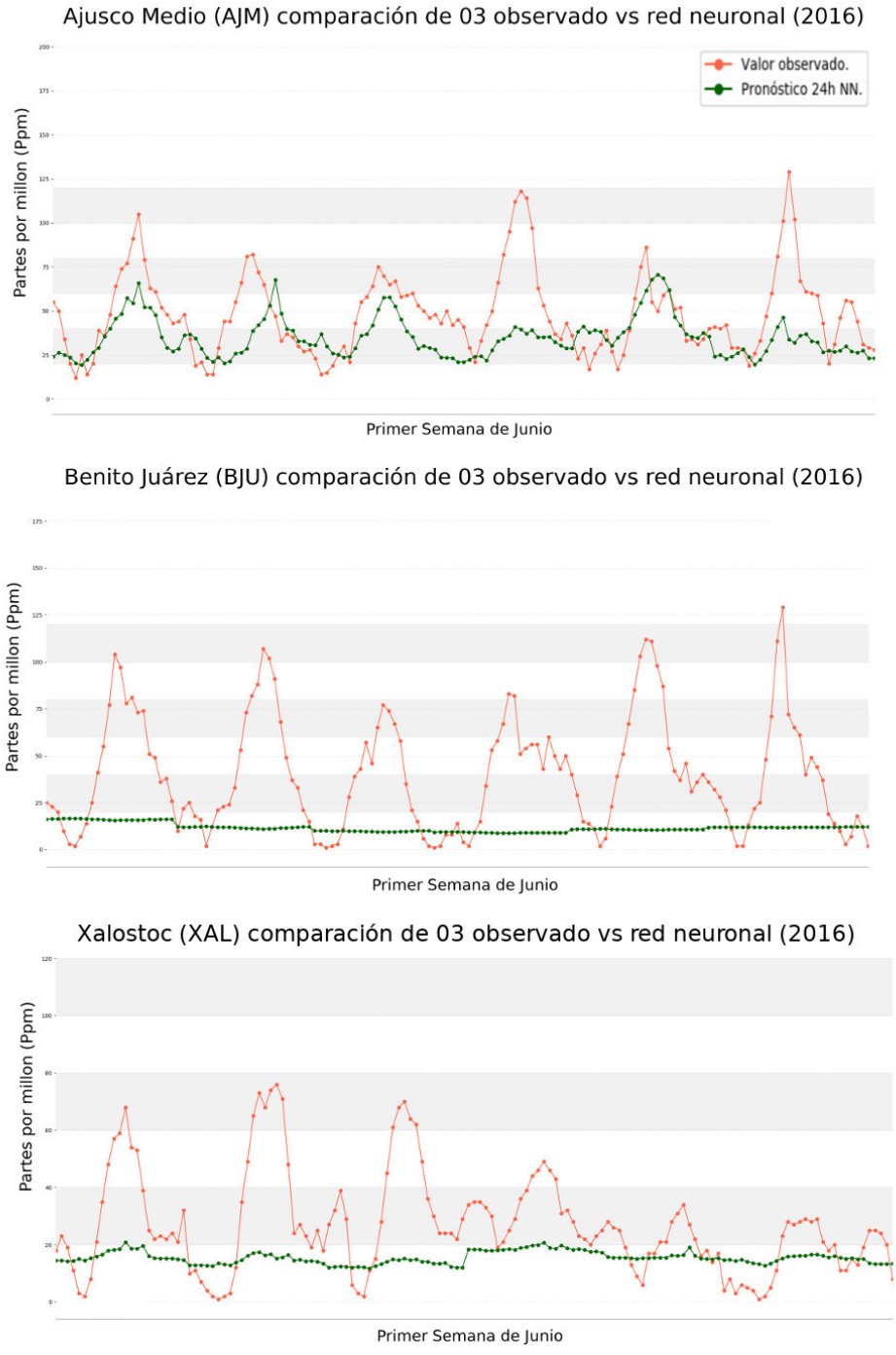

Figura 16: Predicción de las estaciones Ajusco Medio, Benito Juárez y Xalostoc del año 2016 con la configuracion GCC.

15.2. Resultados CM (Datos completos, sin datos meteorológicos, SGD como función de minimización)

Para encontrar solución a este problema se cambió el algoritmo de minimización de error. En la prueba anterior las redes usaron GD como algoritmo de minimización, en la siguiente prueba se cambió por la función SGD. De la misma manera que en la prueba anterior, solamente se le suministró la información de los contaminantes.

Como resultado de cambiar la función de minimización del error se obtienen índices de correlación y de Utheils más altos al que se tenía en la prueba anterior asi como un error menor como se puede ver en las graficas 10 y 11. Ahora se tienen estaciones como Xalostoc y La Presa con un índice de correlación cercano a 1 y un promedio para todas las estaciones de 0.659. Un resultado alentador, pero la mayoría de las estaciones apenas alcanzan un índice de correlación de 0.5, como se puede observar en la figura 17.

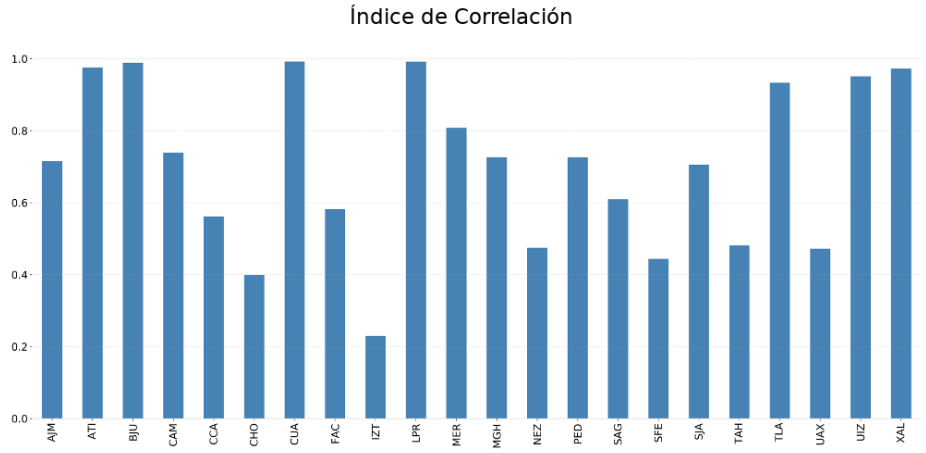

Figura 17: Indice de correlación de la prueba para la configuración CM

A pesar de tener un mejor resultado en el índice de correlación, las gráficas de pre-

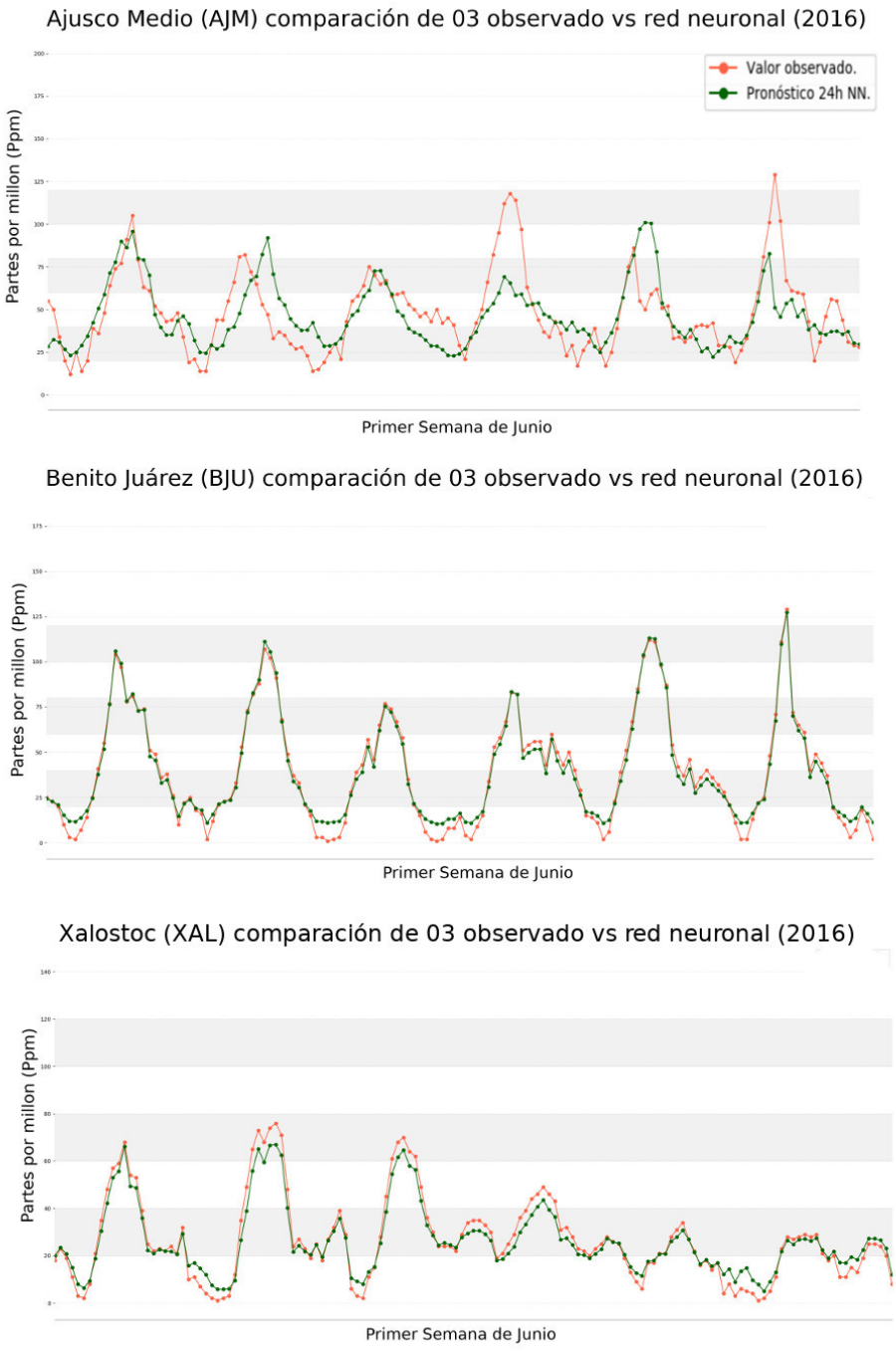

Figura 18: Predicción de las estaciones Ajusco Medio, Benito Juárez y Xalostoc del año 2016 con la configuración de dato CM.

dicción del año 2016 (figura 18) muestran que para la estación Ajusco medio se tiene una predicción más prometedora. La red neuronal ya es capaz de seguir el comportamiento de la emisión de ozono pero aún se está lejos de alcanzar los picos y los valles de la emisión de O_3. Para la estación Benito Juárez se logra un mejor desempeño, pero, existe una sobrestimación en los pico de emisión de O_3 y en los caso de los valles es donde se nota una mala predicción. Para la tercer estación que se tomo para ver los resultados, Xalostoc se tiene que la predicción logra hacer un buen seguimiento, sin embargo hay picos en donde se queda corta la predicción y al igual que en la estación Benito Juárez hay un problema en la predicción de los valles.

15.3. Resultados CC (Datos completos, datos meteorológicos por cuadrantes, SGD como función de minimización)

Ahora que se ha mejorado el pronóstico sin el uso de los datos de meteorología, el paso consecuente es entrenar a la red neuronal tanto con la información de los contaminantes como la información meteorológica. La forma que se suministran los datos de los contaminantes son de manera completa, es decir se dejarán todos los datos. Para cada variable meteorológica se obtiene el cuatro valores por cada hora, dicha configuración está descrita en la sección 12.

El promedio del índice de correlación para esta prueba es de 0.841, todas las estaciones tiene ya un índice de correlación mayor a 0.6. Hay 9 estaciones con un índice de correlación mayor o igual a 0.9, como se puede observar en la figura 19.

A pesar de tener un promedio alto en el índice de correlación el comportamiento de las predicciones por cada estación nos muestra si existe alguna tendencia en el pronóstico (figura 20).

La estación Ajusco medio tiene un índice de correlación de 0.622, la gráfica de pre-

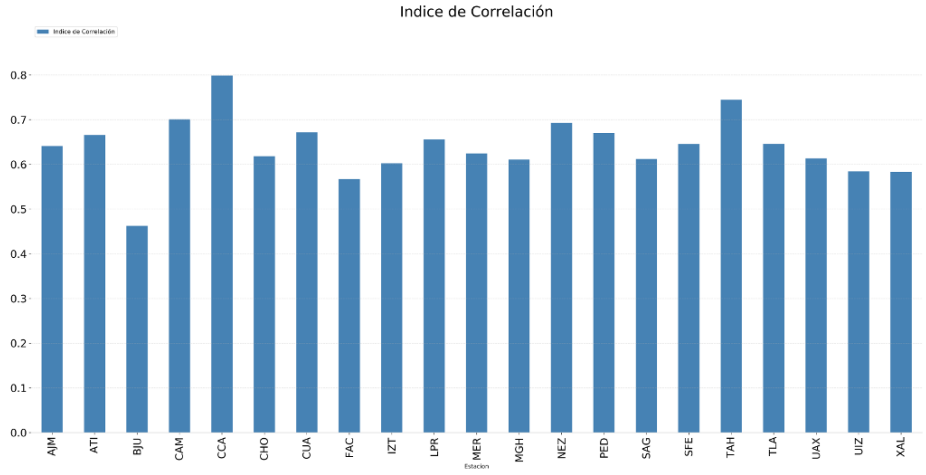

Figura 19: Índice de correlación de la predicción de las redes neuronales para el año 2016 con la información meteorológica (configuración CC de la tabla 7).

dicción deja ver que a la red neuronal de la estación le cuesta trabajo llegar a los picos de las emisiones de O_3 y aunque en menor medida también a los valles. Para la estación de Benito Juárez con un índice de correlación de 0.924, a pesar de tener un índice de correlación alto la gráfica de predicción muestra que la predicción de los picos aún no es la ideal mientras que la predicción de los valles lo hace de mejor manera. Para la estación de la Merced el índice es de 0.990, el mejor de las tres estaciones, la gráfica de predicción muestra que la red neuronal de esta estación hace un buena predicción tanto para los picos y los valles.

Qué pasa con estaciones como Ajusco medio que tiene un índice de correlación de 0.622, si bien es un mejor índice que en la primera prueba aún se está lejos de estaciones como la Merced. En busca de mejorar las estaciones con un índice de correlación bajo se prueban las redes neuronales con diferentes configuraciones de datos, dichas configuraciones están descritas en la tabla 7.

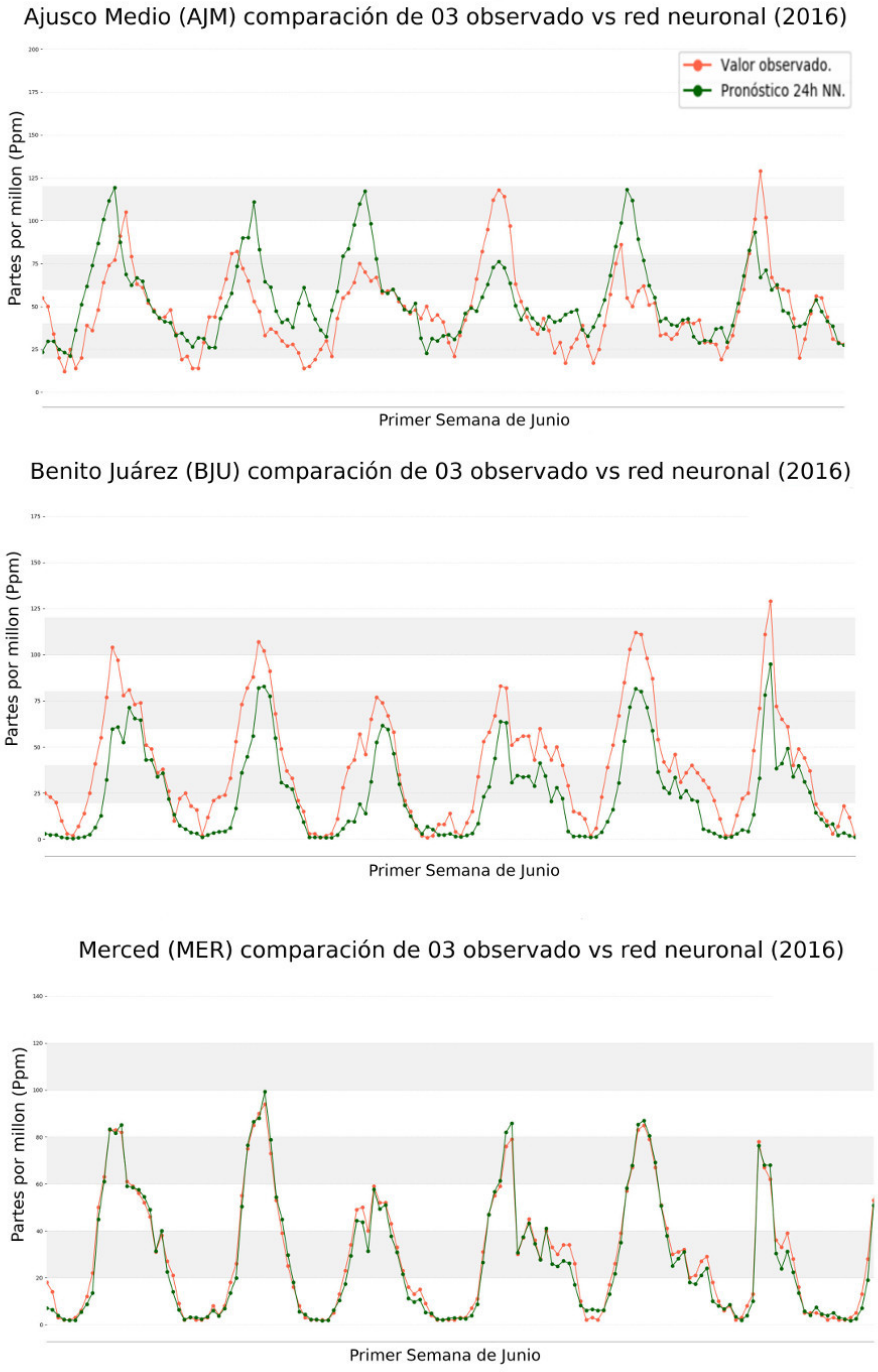

Figura 20: Predicción de las estaciones Ajusco Medio, Benito Juárez y Merced del año 2016 ya con los datos meteorológicos

15.4. Resumen de todas las configuraciones

Las figuras 21 y 22 muestran el resultado de entrenar las redes neuronales con las diferentes configuraciones de datos, ademas en la tabla 15.4 se muestra el resultado de todas las pruebas. Para las redes neuronales entrenadas con Gradient Descent "*GCC* y *GLC*" como función de minimización de error se muestra un bajo índice de correlación de 0.211 y 0.277 y en su contra parte un valor alto en Utheils 0.496 y 0.447, hay que recordar que para Utheils obtener un valor cercano a 0 es mejor mientras que para el índice de correlación se quiere obtener un valor cercano a 1. Este resultado es algo que se esperaba, ya que usar Gradient Descent como función para minimizar el error no es la mejor opción.

Para las configuraciones ya con Stocastic Gradient Descent como función para minimizar el error , pero que no se le suministra la información meteorológica como son "*CM* y *LM*" tienen un índice de correlación de 0.659 y 0.719 mientras que su índice de Utheils es de 0.270 y 0.242 respectivamente, se muestra una mejora considerable en los pronósticos al cambiar la función de minimización, aunque falta de proveer la información meteorología a la red neuronal y ver su comportamiento. En este punto las pruebas dejan ver que la configuración dónde se quitan los datos nulos presenta mejor índice de correlación así como de Utheils, comparando en las que no se remueven.

Para las configuraciones que ya cuentan con los datos de meteorología "*CC*, *CP*, *LC* y *LP*" se tiene un índice de correlación de 0.841, 0.837, 0.841 y 0.831 mientras que el de Utheils los valores son de 0.176, 0.178, 0.180 y 0.183 respectivamente. El proveer de la información meteorológica a la red neuronal mejoró el desempeño de las redes neuronales, dejando un promedio en el índice de correlación de cualquiera de las cuatro configuraciones por arriba de 0.8 mientras que el promedio de Utheils se encuentra por debajo de 0.19. Aunque se tiene resultados positivos y en las cuatro configuraciones

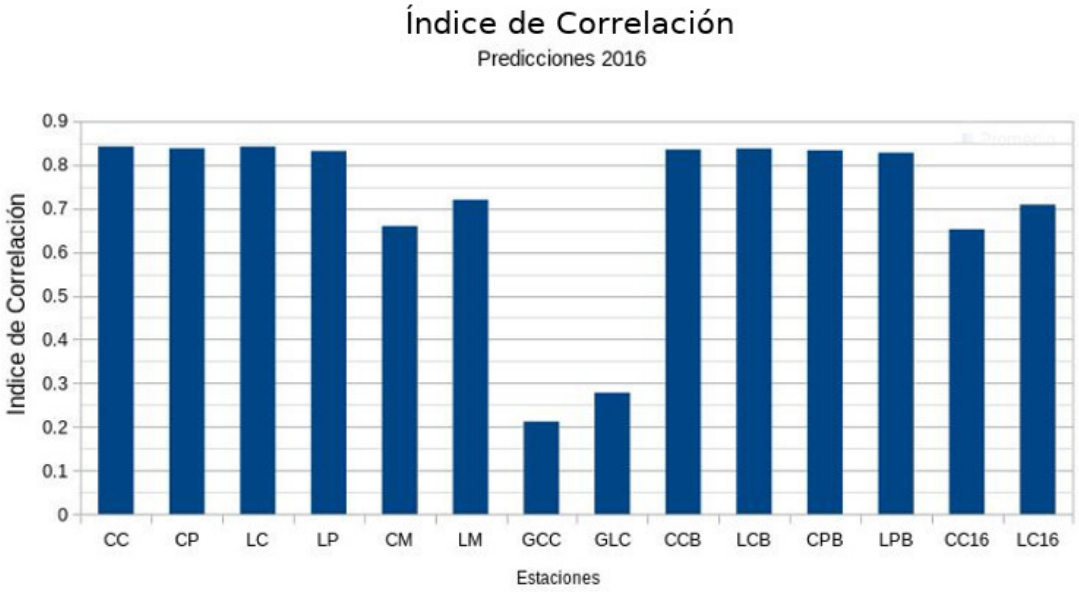

Figura 21: Promedio del índice de correlación de las configuración descritas en la tabla 7.

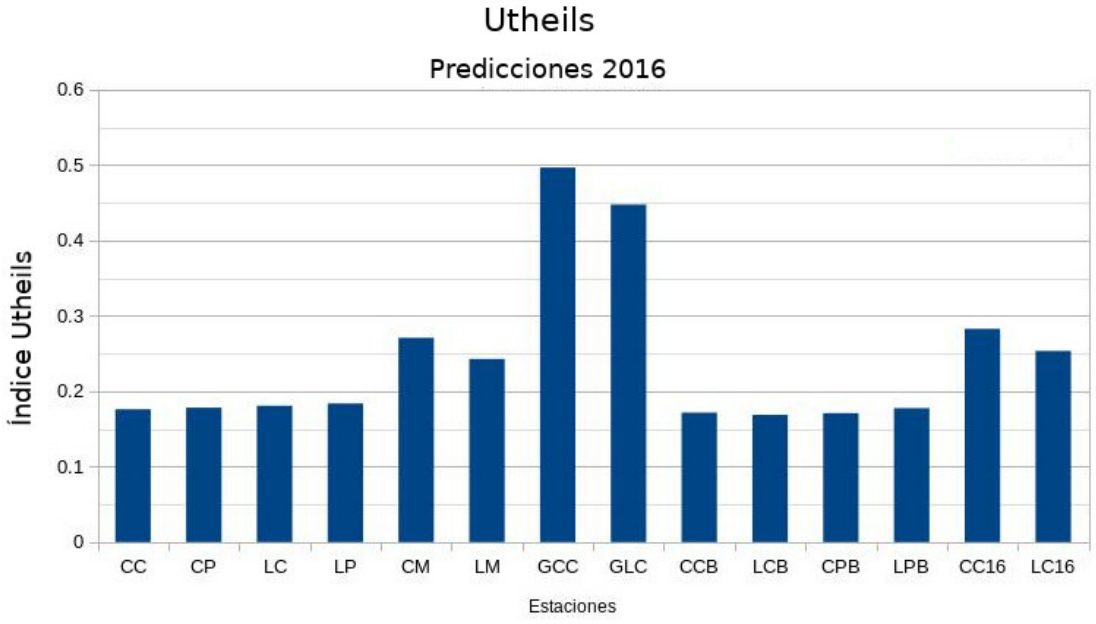

Figura 22: Promedio de Utheils de las configuración descritas en la tabla 7.

resultados muy similares, aplicar bootstrap a estas cuatro configuraciones nos deja ver si alguna de ellas mejora el desempeño de la red neuronal.

Ya con las redes neuronales entrenadas con la configuración de datos aplicando bootstrap, los resultado aunque mejoraron, no dejan de estar muy cercanos a las pruebas anteriores, tenemos que para las cuatro configuraciones "*CCB*, *LCB*, *LCB* y *CPB*" el promedio del índice de correlación de 0.834, 0.837, 0.833 y 0.827, mientras que el de Utheils es de 0.171, 0.168, 0.170 y 0.177 respectivamente, estos resultados nos dejan ver dos configuraciones con el mejor índice de correlación y el de Utheils, tenemos que la configuración *LBP* tiene el mejor índice de correlación de 0.8331 mientras que la configuración *LCB* tiene el mejor índice de Utheil con 0.168.

15.5. Con LCB (Datos limpios, meteorología por cuadrantes y SGD).

Después de probar las 14 configuraciones descritas en la tabla 7 para entrenar la red neuronal, se tomó como configuración principal la *LCB*, siendo la mejor configuración en Utheils, para ver su desempeño general en la figura 23 se muestra el índice de correlación y en la figura 24 se muestra Utheils para el pronóstico del año 2016 de cada estación con la configuración *LCB*, podemos observar que en la mayoría de las estaciones se tiene un índice de correlación mayor a 0.6, solo la estación *AJM* se encuentra por debajo de esta valor. Tenemos que 9 de las 22 estaciones tiene un índice cercano a 1, mientras que en el la gráfica de Utheils (figura 24) observamos que estaciones como SFE, MGH y AJM tiene un valor por arriba de 0.25, siendo estas tres estaciones la que tiene el peor desempeño tanto en el índice de correlación con valores de 0.672, 0.684 y 0.566, mientras que para Utheils son de 0.297, 0.275 y 0.256 respectivamente.

Las 9 estaciones con mejor índice de correlación son ATI, BJU, CUA, LPR, MER, PED, TLA, UIZ y XAL. Se muestra en la figura 25 las gráficas de predicción de las

Configuración	**Índice de correlación**	**Utheils**
CC	0.841	0.176
CP	0.837	0.178
LC	0.841	0.180
LP	0.831	0.183
CM	0.659	0.270
LM	0.719	0.242
GCC	0.211	0.496
GLC	0.277	0.496
CCB	0.834	0.171
LCB	0.837	0.168
CPB	0.833	0.170
LPB	0.827	0.177
CC16	0.652	0.282
LC16	0.708	0.253

Tabla 9: Resultado del índice de correlación y de Utheil para todas las configuraciones

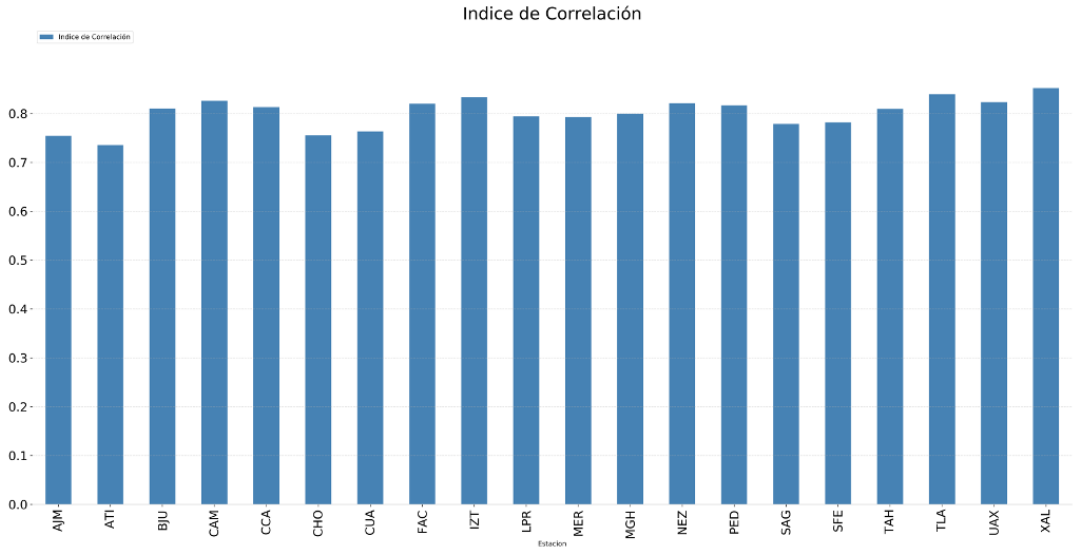

Figura 23: Gráfica con el índice de correlación de cada estación usando la configuración de datos *LCB* en la tabla 7.

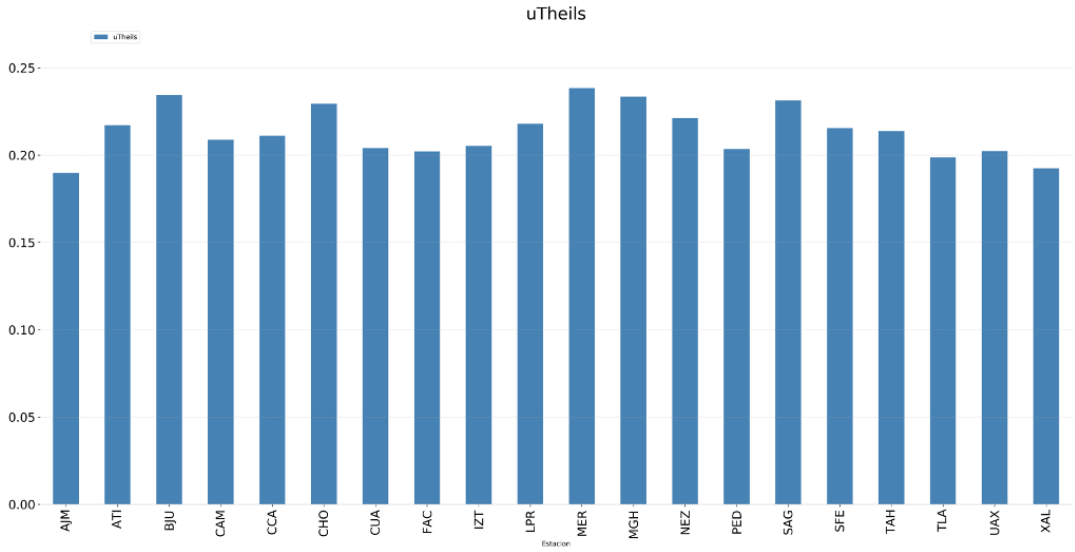

Figura 24: Gráfica con Utheils de cada estación usando la configuración de datos *LCB* en la tabla 7.

estaciones *PED* y *MER* de la primera semana de junio del 2016, además de esta dos estaciones se muestra la gráfica de predicción de la estación *SFE* de igual manera de la

primera semana de junio del 2016.

15.6. Pruebas para el 2017 con LCB (Datos limpios, meteorología por cuadrantes y SGD).

Ahora la configuración de la red neuronal junto con la configuración de los datos se entrena hasta el 2016 y se pronosticar el año 2017, las gráficas de índice de correlación y de Utheils (figuras 26 y 27) muestran que el comportamiento de las estaciones es similar a la prueba del pronóstico de 2016, las 9 estaciones que mostraron buen desempeño en el año 2016 se mantienen, mientras que las de peor desempeño siguen siendo las mismas. Además de los índices se muestran las gráficas de predicción para la primer semana de junio del 2017 para las estaciones XAL, MER y SFE (figura 28).

15.7. Indice de correlación contra cantidad de datos.

A pesar de tener 9 estaciones con un buen índice de correlación y de Utheils. ¿Qué es lo que pasa con las 13 estaciones restantes?, de las 13 estaciones restantes sólo tres tienen valores menores a 0.6 y un índice de correlación y valores mayores de .025 de Utheils, dejando 10 estaciones con un índice entre 0.6 y 0.7. Estas estaciones se encuentran con un índice por arriba de la media, su desempeño en algunos casos es de sobre estimar los picos y los valles de la emisión O_3 ¿Cómo mejorar estas estaciones?, a pesar de tener estaciones que tienen un alto índice de correlación, hay estaciones de las que no se obtiene un buen pronóstico. Una de las razones puede ser la cantidad de datos con la que cada estación es entrenada. Para confirmar o negar este supuesto en la gráfica (figura 29) se muestra una comparación del índice de correlación contra el número de datos con los que es entrenada cada estación.

En la figura 29 observamos que estaciones que cuentan con un número alto de

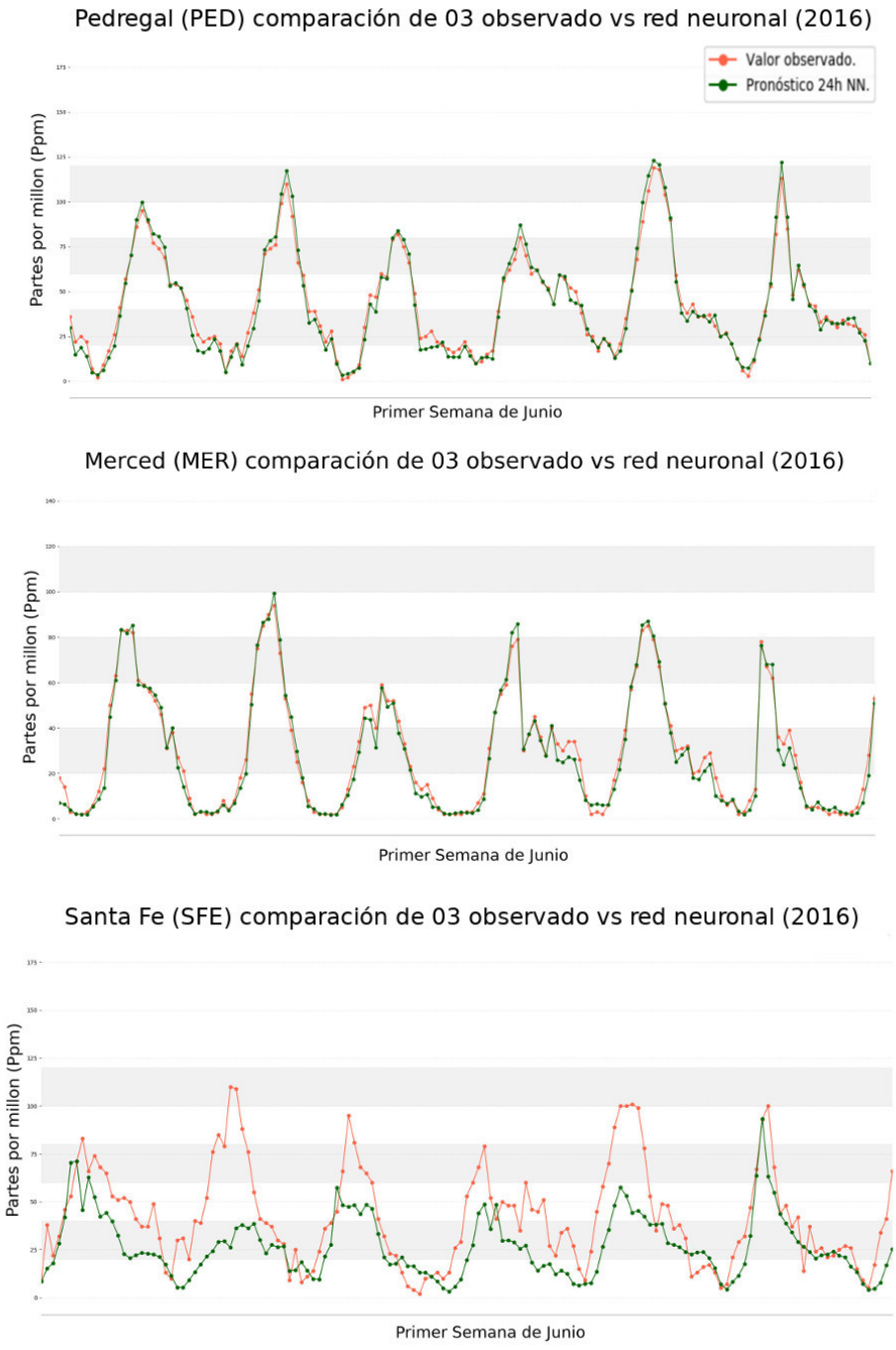

Figura 25: Gráficas de predicción de las estaciones *PED*, *MER* y *SFE* de la primer semana de junio de 2016 hechas con la configuración LCB.

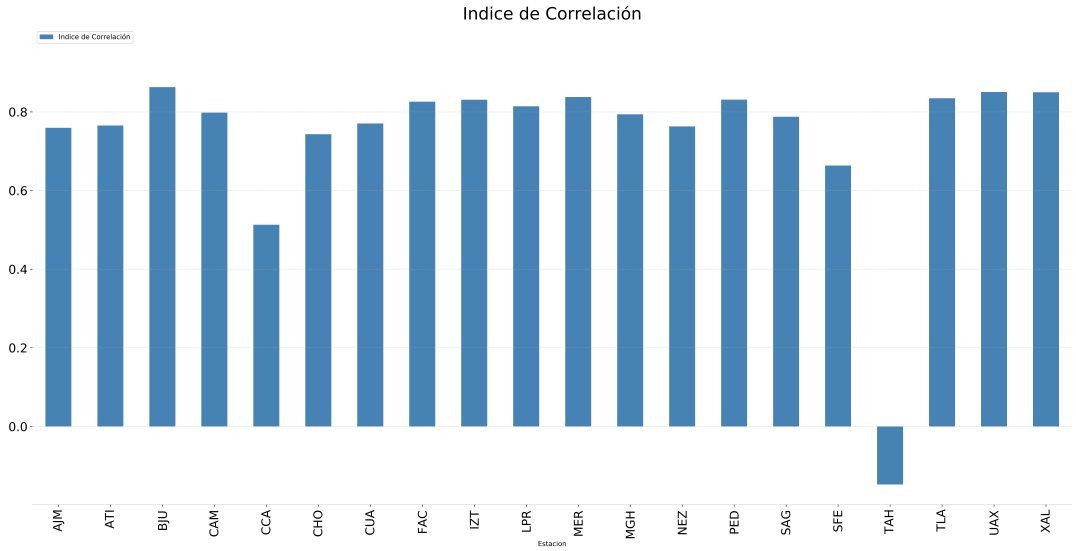

Figura 26: Gráficas del índice de correlación para la predicción del año 2017.

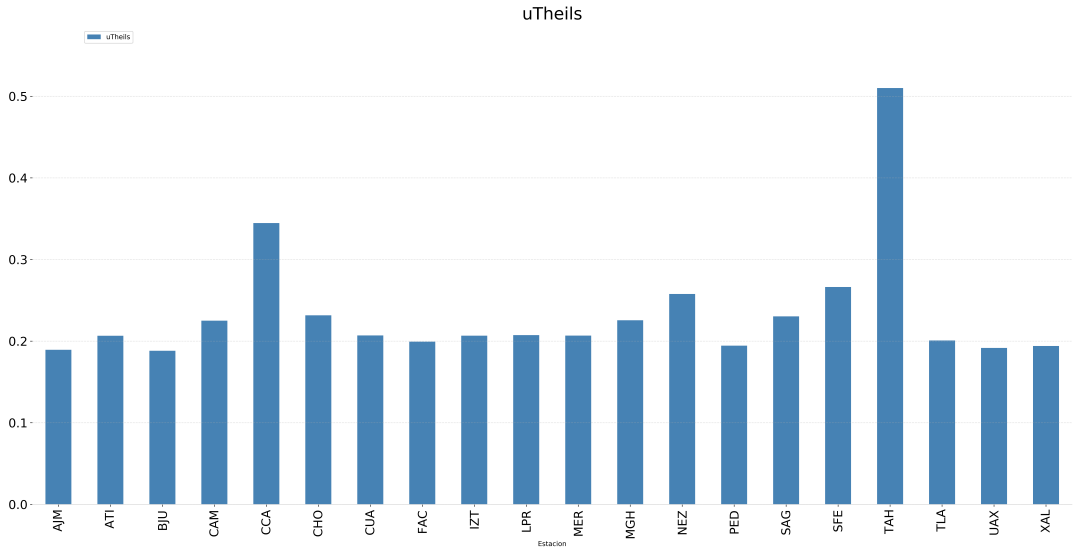

Figura 27: Gráficas del índice de Utheils para la predicción del año 2017.

datos, como es el caso de las estaciones de *XAL* y *UIZ* también tienen un índice de correlación alto. Sin embargo este hecho no ocurre para todas las estaciones, un ejemplo de esto es lo que pasa con la estación de *BJU* que a pesar de tener un bajo número de datos su índice de correlación es alto. Este hecho nos deja ver que no hay una relación

directa entre el número de datos y su índice de correlación. Una investigación más profunda acerca de este comportamiento así como sus posibles soluciones se deja como un trabajo a futuro.

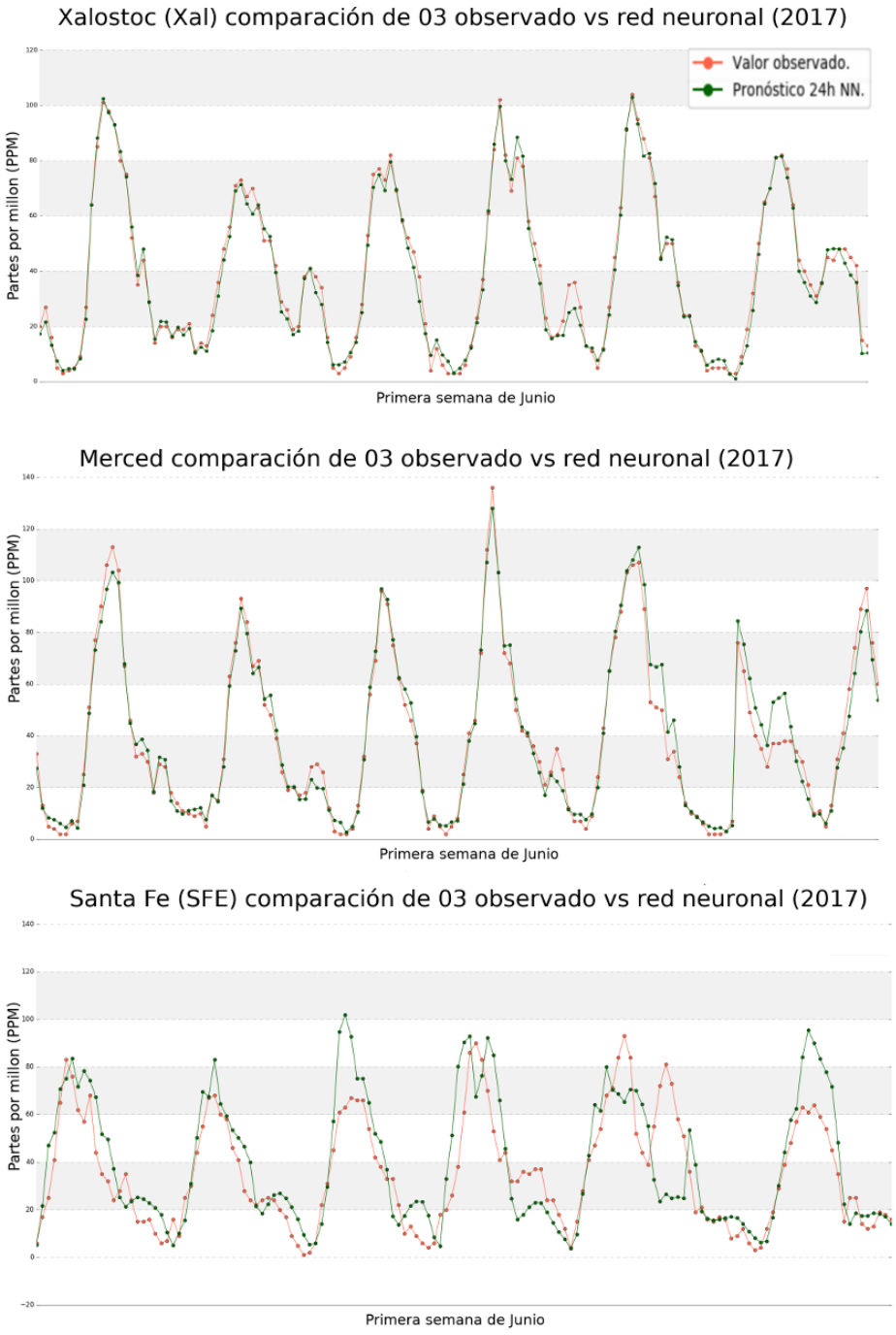

Figura 28: Gráficas de predicción de las estaciones *XAL*, *MER* y *SFE* de la primer semana de junio de 2017.

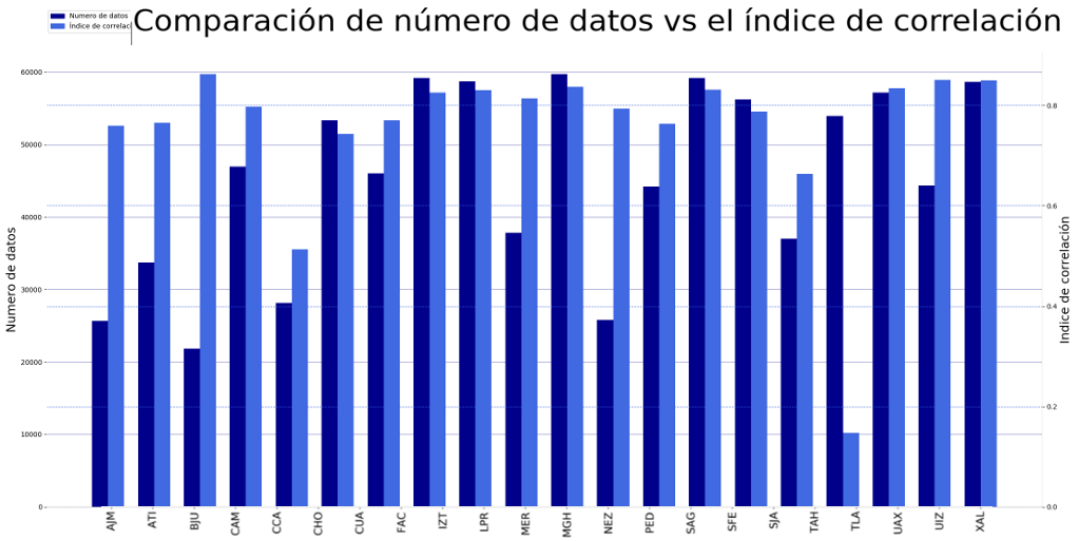

Figura 29: Gráfica de comparación del índice de correlación contra el número de datos de entrenamiento.

Conclusiones

Como resultado de la tesis se realizó un sistema de 22 redes neuronales artificiales, cada una de ellas pertenece a una de 22 estaciones seleccionadas de la RAMA. Las redes neuronales son entrenadas con la información de 7 años de datos tanto de información meteorológica así como de contaminantes. De las 22 estaciones seleccionadas, 9 obtuvieron un índice de correlación mayor a 0.9 mientras y un índice de Utheils menor de 0.2.

Con el desarrollo de este trabajo, podemos concluir que las redes neuronales son una opción para la predicción de O_3, e incluso para otros contaminantes atmosféricos. La selección de una correcta configuración de la red neuronal así cómo los datos y la forma en que se le suministran a la red neuronal, son unos de los problemas que se tiene que atacar y solucionar de manera óptima para poder obtener resultados satisfactorios. La selección de una función de minimización del error es una de las partes más importantes a seleccionar, una mala selección puede provocar que el desarrollo de el sistema se pueda ver afectado al pensar que la falta de datos son el problema y no la mala configuración de la red neuronal.

Cambiar la función de minimización del error así como incluir la meteorología mejoró el índice de correlación promedio. Es necesario evaluar cuál es la precisión de las redes neuronales para pronosticar las contingencias, es decir, en cuántas ocasiones el sistema propuesto predice de forma correcta exceder los límites de contaminación.

Empezar el desarrollo con una red neuronal con una arquitectura simple permite verificar que la construcción de la misma es la óptima y centrarse en la configuración de los datos. Uno de los principales problemas que uno se puede encontrar en el desarrollo de una sistema de redes neuronales es cómo mejorar el desempeño de las mismas. Si la arquitectura de la red neuronal es sencilla, ¿Cómo saber si el hacerla más compleja mejorará el desempeño de la misma? o si acaso el hecho de tener una arquitectura más compleja requerirá de más cantidad de datos o incluso de aumentar el número de características que se le suministran a la red neuronal.

Un hecho particular es lo que pasa en las redes entrenadas con la información de contaminantes dada en 16 cuadrantes "$CC16$ y $LC16$", al parecer dar la información de un mayor número de cuadrantes no ayuda en el pronóstico de la red neuronal, este hecho se lo podríamos atribuir que el número de capas ocultas y el número de neuronas con las que cuenta cada capa de la red no son suficientes para poder propagar la información detallada de la meteorología. Aunque interesante este resultado y sus posibles soluciones, se dejará para trabajo futuro la búsqueda de una mejor configuración.

El resultado de las redes neuronales en el pronóstico de la emisión de O_3 fue satisfactorio en 9 de las 22 estaciones, con un índice de correlación mayor a 0.9, en 6 de estas estaciones el índice de correlación es mayor a 0.98. Aunque en las 13 restantes estaciones, una investigación más profunda acerca de los factores que pueden estar influenciando en la predicción de O_3 es necesaria.

Otro de los resultado interesante, es lo que pasa en la comparación del número de datos contra el índice de correlación (gráfica 29). Este resultado requiere una investigación acerca de lo que sucede con estaciones que tienen un alto número de datos pero un índice de correlación bajo o viceversa ¿qué factores están influenciando este comportamiento?.

Más allá de los resultados obtenidos con las redes neuronales, el trabajo hecho en conjunto al área de las ciencias atmosféricas deja ver que hay un gran campo de

trabajo multidisciplinario entre las ciencias de la computación y otras áreas científicas. Es así como el uso del aprendizaje automático y la enorme cantidad de datos que otras áreas de la ciencias logran recabar con el paso de los años puede mejorar las técnicas convencionales que usan dichas áreas.

Bibliografía

[1] U. E. P. Agency. Code base for the U.S. EPA's Community Multiscale Air Quality Model (CMAQ). For additional background on CMAQ please visit: www.epa.gov/CMAQ, Oct. 2017. original-date: 2016-09-06T13:36:41Z.

[2] Andrew NG. Stochastic Gradient Descent - Universidad de Stanford, 2012.

[3] Y. Bengio. Practical recommendations for gradient-based training of deep architectures. *arXiv:1206.5533 [cs]*, June 2012. arXiv: 1206.5533.

[4] Bradley Efron. Better bootstrap confidence intervals. *Journal of the American Statistical Association*, 82(397):171–185, 1987.

[5] J. Brownlee. Gentle Introduction to the Adam Optimization Algorithm for Deep Learning, July 2017.

[6] M. Cai, Y. Yin, and M. Xie. Prediction of hourly air pollutant concentrations near urban arterials using artificial neural network approach. *Transportation Research Part D: Transport and Environment*, 14(1):32–41, Jan. 2009.

[7] A. CDMX. Dirección de Monitoreo Atmosférico, 2016.

[8] Christos Stergiou and Dimitrios Siganos. Neural Networks, 1996.

[9] M. Cost and i. The discovery of the neuron, Aug. 2006.

[10] Efron Bradley. Efron : Bootstrap Methods: Another Look at the Jackknife, 1977.

[11] Fidel Ramón and Jesús Hernández-Falcón. El Potencial de Acción, Mar. 2005.

[12] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[13] G. Grivas and A. Chaloulakou. Artificial neural network models for prediction of PM10 hourly concentrations, in the Greater Area of Athens, Greece. *Atmospheric Environment*, 40(7):1216–1229, Mar. 2006.

[14] Gustavo Garza and Fernando Aragon. La contaminación atmosférica de la ciudad de México en escala megalopolitana. *Estudios demográficos y urbanos*, 2011.

[15] L. Hrust, Z. B. Klaić, J. Križan, O. Antonić, and P. Hercog. Neural network forecasting of air pollutants hourly concentrations using optimised temporal averages of meteorological variables and pollutant concentrations. *Atmospheric Environment*, 43(35):5588–5596, Nov. 2009.

[16] Humberto Bravo A., G. Roy-Ocotla R., Sanchez A., and R. Torres J. Contaminación atmosférica por ozono en la zona metropolitana de la ciudad de méxico: evolución histórica y perspectivas. *Sección de Contaminación Ambiental, Centro de Ciencias de la Atmósfera, UNAM.*, 1993.

[17] INEGI. Información por entidad. Cuéntame, 2015.

[18] INEGI. Vehículos de motor registrados en circulación, 2015.

[19] D. Jiang, Y. Zhang, X. Hu, Y. Zeng, J. Tan, and D. Shao. Progress in developing an ANN model for air pollution index forecast. *Atmospheric Environment*, 38(40):7055–7064, Dec. 2004.

[20] Kandel, E. R, Schwartz, J. H., Jessell, T. M., and Agud Aparicio. *Principios de neurociencia*. McGraw-Hill Interamericana de España., 4 ediciion edition, 2001.

[21] K. D. Karatzas and S. Kaltsatos. Air pollution modelling with the aid of computational intelligence methods in Thessaloniki, Greece. *Simulation Modelling Practice and Theory*, 15(10):1310–1319, Nov. 2007.

[22] M. Kolehmainen, H. Martikainen, and J. Ruuskanen. Neural networks and periodic components used in air quality forecasting. *Atmospheric Environment*, 35(5):815–825, Jan. 2001.

[23] Larry Hardesty. Explained: Neural networks, 2017.

[24] W. Z. Lu, W. J. Wang, X. K. Wang, Z. B. Xu, and A. Y. T. Leung. Using Improved Neural Network Model to Analyze RSP, NOx and NO2 Levels in Urban Air in Mong Kok, Hong Kong. *Environ Monit Assess*, 87(3):235–254, Sept. 2003.

[25] H. Niska, T. Hiltunen, A. Karppinen, J. Ruuskanen, and M. Kolehmainen. Evolving the neural network model for forecasting air pollution time series. *Engineering Applications of Artificial Intelligence*, 17(2):159–167, Mar. 2004.

[26] J. B. Ordieres, E. P. Vergara, R. S. Capuz, and R. E. Salazar. Neural network prediction model for fine particulate matter (PM2.5) on the US–Mexico border in El Paso (Texas) and Ciudad Juárez (Chihuahua). *Environmental Modelling & Software*, 20(5):547–559, May 2005.

[27] P. Perez and J. Reyes. An integrated neural network model for PM10 forecasting. *Atmospheric Environment*, 40(16):2845–2851, May 2006.

[28] S. J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach (2nd Edition)*. Prentice Hall, December 2002.

[29] A. Russo, F. Raischel, and P. G. Lind. Air quality prediction using optimal neural networks with stochastic variables. *Atmospheric Environment*, 79:822–830, Nov. 2013.

[30] Sebastian Raschka. What's the difference between gradient descent and stochastic gradient descent? - Quora, 2015.

[31] Sebastian Ruder. An overview of gradient descent optimization algorithms, Jan. 2016.

[32] Sedema. La contaminación, tu salud y el transporte, 2016.

[33] SEDEMA. Pronóstico de calidad del aire para la CDMX, 2017.

[34] F. v. Veen. The Neural Network Zoo, Sept. 2016.

[35] P. Viotti, G. Liuti, and P. Di Genova. Atmospheric urban pollution: applications of an artificial neural network (ANN) to the city of Perugia. *Ecological Modelling*, 148(1):27–46, Feb. 2002.

[36] Wilson Casas, Leonard Ortolano, and Ernesto Sánchez Triana. Modelos de calidad del aire, 1995.

[37] O. Zavala. cca_cont: Scripts to analyze data of the contamintants in Mexico City, Oct. 2017. original-date: 2016-06-01T20:27:59Z.

[38] M. Zeiler, M. Ranzato, R. Monga, M. Mao, K. Yang, Q. Le, P. Nguyen, A. Senior, V. Vanhoucke, J. Dean, and G. Hinton. On rectified linear units for speech processing. pages 3517–3521. IEEE, May 2013.