UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
POSGRADO EN CIENCIA E INGENIERÍA DE LA COMPUTACIÓN

ANÁLISIS DE SERIES DE TIEMPO MULTIVARIADAS PARA EL PRONÓSTICO DE CONCENTRACIONES DE CONTAMINANTES DEL AIRE

T E S I S

QUE PARA OPTAR POR EL GRADO DE:

MAESTRO EN CIENCIA E INGENIERÍA DE LA COMPUTACIÓN

PRESENTA:

MARCO ALEXANDER RAMÍREZ SÁNCHEZ

DIRECTORA DE TESIS:

DRA. SUEMI RODRÍGUEZ ROMO

FACULTAD DE ESTUDIOS SUPERIORES CUAUTITLÁN, UNAM

CIUDAD UNIVERSITARIA, CD. MX. OCTUBRE, 2021

Dedicado a mis padres, hermanos y amigos,
quienes siempre me han brindado su apoyo incondicional.

Agradecimientos

A la Universidad Nacional Autónoma de México, por darme la oportunidad de desarrollarme no solo académicamente, sino también personalmente.

Al Consejo Nacional de Ciencia y Tecnología (CONACyT), por haberme apoyado con una beca durante mis estudios de Maestría.

Al Programa de Apoyo a Proyectos de Investigación e Innovación Tecnológica (PAPIIT) de la UNAM IT102920.

A mi directora de tesis la Dra. Suemi Rodríguez Romo, quien me guió con total compromiso en el proceso de desarrollo de este proyecto, compartiéndome en el camino enseñanzas muy valiosas para el crecimiento personal y profesional.

Al Centro de Investigaciones Teóricas de la Facultad de Estudios Superiores Cuautitlán, por todo el apoyo y facilidades durante mi estancia en el posgrado. En especial mis profesores, Mtro. Felipe Vargas Torres, Dr. Óscar Ibáñez, Dr. Ricardo Paramont Hernández (q.e.p.d) y Mtro. Rene Perez Moroyoqui, quienes me brindaron su apoyo y conocimiento para la realización de este trabajo.

A los integrantes del jurado del examen de grado, por aceptar ser parte del jurado que calificó este trabajo y brindar su apoyo y compromiso.

Resumen

Debido a todos los efectos negativos que la contaminación del aire provoca, el control de la contaminación se convierte en una tarea que preocupa a investigadores de todo el mundo. Diferentes estaciones de monitoreo de la calidad del aire en la Zona Metropolitana del Valle de México, adquieren mediciones de diferentes contaminantes (series de tiempo), las cuales se utilizan para comprender la influencia de la contaminación del aire y sus tendencias. Esto se logra mediante pronósticos que nos permitan anticipar los niveles de concentración de los contaminantes. Por este motivo el pronóstico se ha convertido en una tarea base para el control de la contaminación [1].

Sin embargo, antes de aplicar algún método para hacer el pronóstico de series de tiempo, es necesario preparar la información obtenida de un conjunto de datos para eliminar aquellos errores que imposibiliten un buen análisis. Uno de los principales problemas sucede cuando debido a algún mal funcionamiento o perturbación durante la adquisición de datos (p. ej falla de sensores) se presentan datos faltantes, es decir, instancias de la serie de tiempo con ausencia de información. Este problema es inevitable en cualquier serie de tiempo del mundo real y debe solucionarse, de lo contrario se reduce la eficiencia y se dificulta el análisis [2].

En este trabajo se utiliza el modelo de aprendizaje profundo MTS-GAN para realizar la imputación de datos faltantes a series de tiempo multivariadas con un buen nivel de precisión, incluso mejor que varios modelos del estado del arte reportadas a la fecha.

Para el pronóstico de series de tiempo, los métodos pueden dividirse en tres categorías principales: métodos estadísticos, métodos numéricos y métodos de inteligencia artificial. Dentro de este última categoría existe un modelo de redes neuronales conocido como redes neuronales recurrentes (RNN por sus siglas en inglés "Recurrent Neural Network") que ha mostrado tener un buen desempeño para modelar secuencias, al capturar la dependencia temporal no lineal entre el valor futuro predicho y los valores anteriores.

Muchas arquitecturas más complejas han sido desarrolladas a partir de las RNN y han mostrado tener mayor efectividad y eliminar algunos problemas como el desvanecimiento de gradiente. Algunos de estos modelos son: LSTM (Long Short-Term Memory),

GRU (Gated Recurrent Unit), BiLSTM (Bidirectional Long Short-Term Memory), entre otras.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Presentación

Como sabemos, el aire es un componente básico para la vida sobre la Tierra, se encuentra conformado por una mezcla de gases (principalmente N_2, O_2, Ar y CO_2) que constituyen la atmósfera y que son necesarios para que los seres vivos puedan realizar funciones vitales, por ejemplo el O_2 en la respiración de los humanos y el CO_2 como base de la fotosíntesis en las plantas.

La contaminación del aire sucede cuando algunas sustancias y partículas presentes en la atmósfera exceden ciertos niveles de concentración, modificando su composición natural. Este fenómeno ocurre en las capas más bajas de la atmósfera; la estratósfera y la tropósfera, siendo esta última la capa en donde habitan los seres humanos y otros organismos [3].

Las altas concentraciones de contaminantes del aire traen efectos perjudiciales para la salud de los seres vivos, particularmente en el caso de los seres humanos pueden ser enfermedades respiratorias, enfermedades cardiovasculares y cáncer, entre otros [3]. Además la contaminación del aire tiene un gran impacto sobre el medio ambiente, provocando problemas como el cambio climático y calentamiento global [1].

Debido a todos los efectos negativos que la contaminación del aire provoca, el control de la contaminación se convierte en una tarea que preocupa a investigadores de todo el mundo. Ya que los contaminantes no solo se concentran en las áreas donde son emitidos, sino que también pueden dispersarse a otras regiones, incluso viajar grandes distancias y ocasionar daños a escala global [4].

La Secretaría del Medio Ambiente (SEDEMA) se ha encargado de la implementación de distintas estaciones de monitoreo de calidad del aire en distintos puntos de la Zona Metropolitana del Valle de México (ZMVM), con la finalidad de registrar los

niveles de contaminación durante las 24 horas del día.

Los conjuntos de datos obtenidos son publicados por la SEDEMA, donde proporciona las mediciones de 9 contaminantes (CO, NO, NO_2, NO_X, O_3, PM_{10}, $PM_{2.5}$, PMCO y SO_2) en estaciones ubicadas en diferentes puntos de la ZMVM. Además existe otro conjunto con mediciones meteorológicas como: velocidad de viento, dirección del viento, temperatura, etc. Estas mediciones son registradas cada hora durante los 365 días del año (en algunos casos desde el año 1986).

En dicha zona la contaminación del aire es generada principalmente por el transporte, la industria y el uso habitacional [5]. En particular el ozono (O_3) es un gas muy complicado, ya que se considera contaminante o no, según la capa de la atmósfera en la que se encuentre:

- Ozono estratosférico: Conforma lo que conocemos como capa de ozono y se encuentra a una altura aproximada de 20 km, esta capa protege al planeta de los rayos ultravioleta (UV), los cuales traen efectos nocivos para los humanos, animales y plantas.
- Ozono troposférico: Se encuentra a nivel de la superficie, se produce cuando los óxidos de nitrógeno (NO_X) y los compuestos orgánicos volátiles (COV) reaccionan en la atmósfera en presencia de radiación solar [1]. Debido a sus propiedades químicas altamente reactivas, el ozono troposférico es dañino para la vegetación, los materiales y la salud humana [6].

Las mediciones de O_3 obtenidas por las estaciones de monitoreo de calidad del aire de la ZMVM corresponden a ozono troposférico al cual desde ahora simplemente referiremos como ozono (O_3) y que será el principal compuesto a analizar en este trabajo dadas sus particularidades como un contaminante.

Además de los compuestos precursores del ozono (NO_X y COV), los cuales diariamente son capaces de generar altos niveles de este compuesto, existen algunas condiciones climáticas que favorecen su producción:

- Elevada luz solar
- Altas temperaturas
- Poca humedad
- Baja velocidad del viento

Los datos adquiridos (series de tiempo) por las estaciones de monitoreo de la calidad del aire, se utilizan para comprender la influencia de la contaminación del aire y sus tendencias. Esto se logra mediante pronósticos, extendiendo los valores históricos

[1] Los motores de combustión interna son los principales emisores de NO_X en la ZMVM.

disponibles hacia el futuro. Un buen pronóstico permite anticipar los niveles de concentración de los contaminantes, para que así en caso de que estos niveles sean elevados, se tomen las medidas pertinentes para normalizarlos y resguardar la integridad de la población y medio ambiente. Por este motivo el pronóstico se ha convertido en una tarea base para el control de la contaminación [1].

Sin embargo, antes de aplicar algún método para hacer el pronóstico de series de tiempo, es necesario preparar la información obtenida de un conjunto de datos para eliminar aquellos errores que imposibiliten un buen análisis. Uno de los principales problemas sucede cuando debido a algún mal funcionamiento o perturbación durante la adquisición de datos (p. ej falla de sensores) se presentan datos faltantes, es decir, instancias de la serie de tiempo con ausencia de información. Este problema es inevitable en cualquier serie de tiempo del mundo real y debe solucionarse, de lo contrario se reduce la eficiencia y se dificulta el análisis [2].

Existe una gran variedad de métodos para manejar el problema de los datos faltantes, los cuales van desde simplemente eliminar aquellas filas en las que falta información, hasta la utilización de modelos complejos para imputar la información. En este trabajo se utiliza el modelo de aprendizaje profundo MTS-GAN, el cual utiliza redes convolucionales profundas de múltiples canales dentro de una red generativa antagónica para modelar la distribución de una serie de tiempo, y con ello, realizar la imputación de datos faltantes a la serie de tiempo multivariada con un buen nivel de precisión, incluso mejor que varios modelos del estado del arte reportadas a la fecha.

Para el pronóstico de series de tiempo, los métodos pueden dividirse en tres categorías principales: métodos estadísticos, métodos numéricos y métodos de inteligencia artificial. Dentro de este última categoría existe un modelo de redes neuronales conocido como redes neuronales recurrentes (RNN por sus siglas en inglés "Recurrent Neural Network") que ha mostrado tener un buen desempeño para modelar secuencias, al capturar la dependencia temporal no lineal entre el valor futuro predicho y los valores anteriores.

Muchas arquitecturas más complejas han sido desarrolladas a partir de las RNN y han mostrado tener mayor efectividad y eliminar algunos problemas como el desvanecimiento de gradiente. Algunos de estos modelos son: LSTM (Long Short-Term Memory), GRU (Gated Recurrent Unit), BiLSTM (Bidirectional Long Short-Term Memory), entre otras.

En este trabajo se tiene como propósito principal la imputación de datos faltantes en las series de tiempo multivariadas, y una vez reconstruida la series de tiempo hacer pronóstico sobre los niveles de concentración de ozono y las demás variables que componen la serie. La selección de estas variables adicionales se hará de acuerdo a la relación que tengan con la presencia de este contaminante, con el propósito de realizar un pronóstico más preciso. Además, para realizar el pronóstico, se propone utilizar al-

gunas de las arquitecturas más efectivas de redes neuronales y aprendizaje profundo.

1.2. Hipótesis

Es posible mejorar la imputación de datos en las series de tiempo multivariadas de contaminantes del aire en la ZMVM, utilizando un modelo de aprendizaje profundo que aprenda la distribución original de la serie de tiempo. Además, la serie de tiempo reconstruida con este modelo junto con modelos de redes neuronales recurrentes, ofrecerían mejores resultados que series reconstruidas con métodos más tradicionales (como imputar valores medios o KNN) ya que sus predicciones son más precisas.

1.3. Objetivo general

Solucionar el problema de los datos faltantes en las series de tiempo multivariadas de concentración de O_3, temperatura y humedad relativa en diferentes estaciones de la ZMVM, haciendo uso del modelo de aprendizaje profundo MTS-GAN [2]. Los datos imputados deben seguir la distribución original de la serie de tiempo con un buen nivel de precisión para posteriormente hacer predicciones con la utilización de redes neuronales recurrentes.

1.4. Objetivos particulares

1. Formar series de tiempo multivariadas con los niveles de concentración de ozono, temperatura y humedad relativa (preparación de datos).
2. Implementar el modelo de aprendizaje profundo MTS-GAN para imputar los datos faltantes en las series de tiempo multivariadas. Comparar su desempeño frente a otros modelos de imputación de datos más comunes.
3. Implementar modelos de redes neuronales recurrentes para realizar el pronóstico de las series de tiempo reconstruidas en el punto anterior.
4. Comparar los resultados de los pronósticos de series de tiempo reconstruidas con el modelo MTS-GAN contra series de tiempo reconstruidas con otros métodos de imputación más comunes.

1.5. Motivación

Como se ha mencionado existen varios modelos para realizar el pronóstico de series de tiempo que cada vez hacen un mejor trabajo. Sin embargo, a menudo se le da poca relevancia al problema de los datos faltantes y se emplean métodos de imputación sencillos o simplemente se opta por eliminar las filas con ausencia de datos. Pero tomar este tipo de acciones puede dar pie a significativas pérdidas de información, que podrían modificar la distribución original de la serie, lo que es poco deseable al momento de hacer pronósticos para el control de la contaminación del aire. Es por ello que la motivación principal de este trabajo es realizar la imputación de datos con un modelo que nos asegure reconstruir la serie de tiempo siguiendo la distribución original y así obtener un mejor nivel de precisión al momentos de realizar su pronóstico.

Capítulo 2

Marco Teórico

2.1. Series de Tiempo

Una serie de tiempo es un conjunto de datos secuenciales que se obtienen al registrar los valores de una característica (univariada), o varias (multivariada) a lo largo de un periodo de tiempo. Las series de tiempo se encuentran presentes en campos como economía, ingeniería, ciencias sociales, medicina, meteorología, física, solo por mencionar algunos. Algunos ejemplos de series de tiempo son: la población de un país década por década durante el siglo anterior, las ganancias mensuales de una empresa a lo largo de tres años, la producción diaria en una fábrica durante un mes. Además los valores son registrados en intervalos de tiempo regulares (cada hora, cada día, cada mes, etc.), pero también pueden ser no regulares. En la Figura 2.1 se muestra el ejemplo de una serie de tiempo sobre mediciones de temperatura mínima diaria.

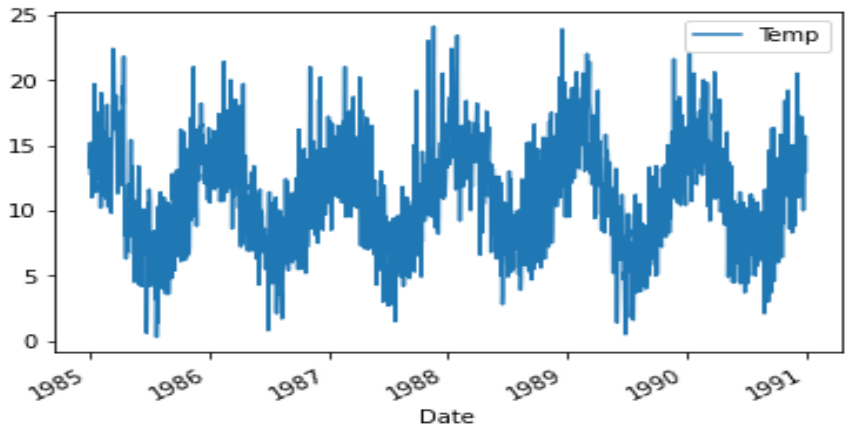

Figura 2.1: Temperaturas mínimas diarias entre 1985 y 1991 en la ciudad de Melbourne, Australia.

2.1.1. Series de Tiempo Univariadas (UTS)

Suponiendo que x es una variable observada en los instantes $t = (1, 2, 3, ..., L)$ donde L es el número de observaciones de la serie de tiempo, es decir, la longitud de la serie. Una serie de tiempo univariada (UTS, por sus siglas en inglés) está definida por $x_1, x_2, x_3, ..., x_L$. Estas observaciones pueden representarse como un vector columna de la siguiente manera:

$$\mathbf{X} = [x_1, ..., x_i, ..., x_L]' \in \mathbb{R}^{L \times 1} \tag{2.1}$$

donde x_i es la i-ésima observación de $\mathbf{X}$.

2.1.2. Series de Tiempo Multivariadas (MTS)

Cuando existen series de tiempo con más de una variable relacionadas, estamos tratando con series de tiempo multivariadas (MTS, por sus siglas en inglés). Frecuentemente trabajar con MTS puede ayudar a realizar un análisis más efectivo, ya que las variables que las componen suelen tener fuertes correlaciones, lo cual permite entender mejor el comportamiento de cada una de las variables y realizar un análisis más preciso.

Dada una MTS con V dimensiones, observada en los instantes $t = (1, 2, 3, ..., L)$, está definida por:

$$\mathbf{X} = [x_1, ..., x_i, ..., x_L]' \in \mathbb{R}^{L \times V} \tag{2.2}$$

donde x_i es la i-ésima observación de $\mathbf{X}$, y x_i^j es la j-ésima variable de x_i. Las observaciones pueden representarse como una matriz de la siguiente manera:

$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1V} \\ x_{21} & x_{22} & \cdots & x_{2V} \\ \vdots & \vdots & & \vdots \\ x_{L1} & x_{L2} & \cdots & x_{LV} \end{bmatrix} \tag{2.3}$$

2.1.3. Componentes de las Series de Tiempo

En una serie de tiempo existen cuatro componentes básicos, que son: tendencia, estacionalidad, ciclicidad y movimientos irregulares. La Figura 2.2 ilustra cada componente sobre una serie de tiempo. Estos componentes proporcionan a la serie los cambios durante un intervalo de tiempo. Identificar estos componentes es útil para poder analizar el comportamiento de la serie y con base a ello realizar predicciones. Estos componentes son:

Tendencia: Este componente representa el movimiento a largo plazo de la serie de tiempo. La tendencia caracteriza el cambio gradual que provoca el crecimiento o

decrecimiento de la serie, aunque también puede suceder que permanezca igual en un cierto intervalo de tiempo. En ocasiones se puede representar con una línea recta o una curva suavizada. Algunos ejemplos de factores que afectan la tendencia de una serie de tiempo son: cambios en la población de algún país, cambios en los ingresos de una empresa, cambios en el nivel de educación, etcétera.

Estacionalidad: La estacionalidad de una serie de tiempo muestra la variación debida a movimientos recurrentes a lo largo de la serie de tiempo que suceden de un periodo estacional al siguiente, las repeticiones pueden tener mayor o menor intensidad y los periodos estacionales pueden estar definidos por día, mes, año, etcétera. Por ejemplo, una tienda de ropa registra sus ventas y observa que las ventas se mantienen casi constantes a lo largo del año y se registran ventas mayores durante el último trimestre, esto sucede año tras año, así que se puede ver claramente la repetición de un año a otro.

Ciclicidad: A menudo las series de tiempo presentan desviaciones de la tendencia provocados por factores distintos a la estacionalidad, a estas desviaciones se les conoce como ciclicidad y generalmente se producen durante un intervalo de tiempo extenso. Este componente resulta difícil de determinar con precisión y a menudo se considera parte de la tendencia, no obstante, se puede obtener una estimación de la ciclicidad al eliminar los componentes de tendencia y estacionalidad.

Movimiento irregular: Este componente es provocado por factores a corto plazo, imprevisibles y no recurrentes que se presentan en la serie de tiempo, es decir, es un movimiento aleatorio e impredecible no descrito por ninguno de los componentes anteriores.

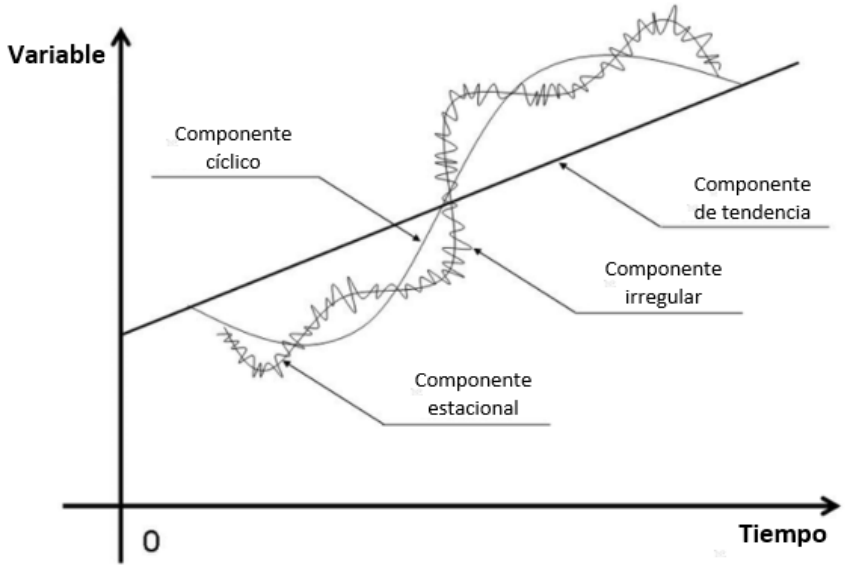

Figura 2.2: Componentes de una serie de tiempo.

2.2. Datos faltantes

Una de las dificultades que existen al analizar las series de tiempo, es el problema de los datos faltantes, que son ausencia de observaciones debido, esto sucede debido a algún mal funcionamiento o disturbio durante la adquisición de datos. Algunos ejemplos de esto sería una avería mecánica en un sensor que captura las mediciones de consumo eléctrico o que en una encuesta algún participante se niegue a dar alguna respuesta. Los datos perdidos traen complicaciones a la hora de realizar el análisis, ya que la serie de tiempo se encuentra incompleta y esto provoca un sesgo debido a la diferencia entre los datos completos y los datos perdidos [2].

Existen una gran cantidad de métodos para solucionar el problema de los datos perdidos en series de tiempo, estos pueden clasificarse en dos enfoques distintos: métodos de eliminación y métodos de imputación [2]. Los métodos de eliminación consisten en remover aquellas observaciones en donde hay datos perdidos. Cuando la cantidad de datos perdidos es demasiado grande, este tipo de métodos pueden resultar contraproducentes, ya que provocan que la serie de tiempo pierda demasiada información útil. Por otra parte, los métodos de imputación llenan las observaciones con datos perdidos aplicando algún algoritmo que se basa en la información disponible, dicho algoritmo tiene como objetivo aproximarse en mayor medida a los datos reales que no se pudieron adquirir.

2.2.1. Métodos de eliminación

Dentro de los métodos de eliminación se encuentran:

- Eliminación por lista: Elimina todos los casos en los que existen datos faltantes, dejando únicamente los casos con datos completos.
- Eliminación por pares: Elimina los casos que tienen datos faltantes en variables involucradas en algunas operaciones.

2.2.2. Métodos de imputación

Existen varios métodos de imputación diferentes. De acuerdo con [2] podemos clasificarlos en tres distintos tipos:

1. Métodos de imputación basados en datos simples: Son los métodos más tradicionales que emplean estadística y algoritmos simples. Algunos de estos métodos son:
    - Imputación de media/mediana/moda: Consiste en llenar aquellos valores perdidos con la media, mediana o moda de la variable de la serie de tiempo. Su principal desventaja es que pueden introducir sesgo y cambiar las características de los datos originales como varianza y covarianza. Puede utilizarse en una MTS.
    - Hot deck: Consiste en rellenar datos faltantes con una respuesta observada de una unidad similar. Este método no puede ser utilizado en MTS.
    - Cold deck: Remplaza un valor faltante con un valor constante de una fuente externa. Al igual que Hot deck no puede usarse en MTS.
2. Métodos de imputación basados en modelos: Métodos de imputación que incluyen modelos estadísticos más complejos. A continuación se mencionan algunos métodos basados en modelos:
    - Interpolación: Obtienen los datos faltantes utilizando valores anteriores y sucesivos. Tienden a fallar cuando existen muchos datos faltantes consecutivos en una variable de una MTS.
    - Regresión: Utilizan un conjunto de variables para predecir los valores faltantes en otro conjunto de variables. Estos métodos no pueden aplicarse cuando existe información faltante en todas las variables de una MTS.
    - K-Vecinos Cercanos (KNN, por sus siglas en inglés): Utiliza medidas de similitud para encontrar las muestras del conjunto de datos más similares a la muestra incompleta e imputarle los datos faltantes. Este método es computacionalmente ineficiente cuando se trata con un conjunto de datos muy grande.

3. Métodos de imputación basados en aprendizaje profundo: Este tipo de método utiliza modelos de aprendizaje profundo que han sido propuestos en años recientes para realizar la imputación de datos faltantes. Estos métodos son una mejor elección para la imputación de datos en MTS. Algunos de estos modelos son:

    - Redes Neuronales Recurrentes (RNN, por sus siglas en inglés): Modelo de aprendizaje supervisado caracterizado por tratar con datos secuenciales. El modelo inicializa los datos faltantes para imputarlos y los actualiza cuando se entrena.
    - Codificadores Automáticos de Eliminación de Ruido (DAE, por sus siglas en inglés): Modelo no supervisado que reconstruye la entrada de su versión corrupta con la mayor fidelidad posible.
    - Redes Generativas Antagónicas para Series de Tiempo Multivariadas (MTS-GAN, por sus siglas en inglés): Modelo generativo que modela bastante bien la distribución de una MTS, para generar los datos faltantes.

Este trabajo se centrará en los métodos de imputación, en particular el modelo MTS-GAN, ya que se considera la mejor opción, debido a que tienen un buen desempeño, incluso cuando existe una gran cantidad de datos faltantes.

2.3. Pronóstico de Series de Tiempo

El pronóstico de series de tiempo, consiste en utilizar las observaciones actuales y pasadas disponibles de una serie de tiempo para predecir su valor en el futuro con la mayor precisión posible [7]. Por lo tanto, el pronóstico de series de tiempo es de gran importancia en diversos campos y en muchas situaciones, algunos ejemplos de situaciones donde se requiere el pronóstico de series de tiempo descritos por R. Hyndman en [8] son: "Decidir si construir otra planta de energía en los siguientes cinco años requiere pronóstico de la demanda, programar al personal de un call center requiere pronóstico del volumen de llamadas, almacenar un inventario requiere pronóstico de los requisitos de stock".

Un modelo para el pronóstico de series de tiempo toma la información numérica disponible del pasado para intentar predecir sus próximos valores, capturando algunos patrones pasados que pueden continuar en el futuro. Podemos clasificar estos modelos en tres categorías clásicas: métodos estadísticos, métodos numéricos y métodos de inteligencia artificial [1]. Cada modelo ofrece distintas ventajas y puede ser muy simples o muy complejos. La elección de alguno de ellos depende de la cantidad de datos disponibles. A continuación se describen algunos de los modelos más utilizados:

- Autoregresivo Integrado de Media Móvil (ARIMA, por sus siglas en inglés): Uno de los métodos estadísticos más populares para el pronóstico de series de tiempo. El modelo utiliza variaciones y regresiones para obtener la relación entre una

observación y un cierto número de observaciones anteriores y así, predecir valores futuros. La notación estándar del modelo es ARIMA(p, d, q) donde p (orden retraso), d (grado de diferenciación) y q (orden de la media móvil) son valores enteros que especifican el modelo ARIMA que se está utilizando. El modelo puede describirse como:

$$y'_t = c + \phi_1 y'_{t-1} + ... + \phi_p y'_{t-p} + \theta_1 \epsilon_{t-1} + ... + \theta_q \epsilon_{t-q} + \epsilon_t \tag{2.4}$$

en donde y'_t y es la serie diferenciada. Del lado derecho encontramos los valores de y'_t y errores. Los valores $\phi_1 ... \phi_p$ son parámetros correspondientes a la autoregresión del modelo, $\theta_1 ... \theta_q$ son parámetros correspondientes a las medias móviles del modelo y ϵ_t el término de error.

- Máquina de Soporte Vectorial (SVM, por sus siglas en inglés): Modelo de aprendizaje automático, ideado principalmente para resolver problemas de clasificación, pero fácilmente adaptado para resolver problemas de clasificación y problemas de regresión. A grandes rasgos una SVM representa los puntos de la muestra en el espacio y los separa mediante un hiperplano con la mayor separación posible, dividiendo así las clases, cuando se ingresan nuevas muestras estas se clasificarán de acuerdo a la clase en donde se encuentren.

  Cabe remarcar que cuando el número de muestras de entrenamiento es grande los gastos de cómputo y tiempo aumentan significativamente. Además de ser un modelo muy sensible a los datos faltantes [1].

- Redes Neuronales Artificiales (ANN, por sus siglas en inglés): Son un modelo matemático inspirado en el cerebro, utilizado para procesar información, a través de la conexión interna entre múltiples nodos. Estos modelos tienen la capacidad de auto aprender. Algunos de los modelos de ANN utilizados para el pronóstico de series de tiempo son: perceptrón multicapa (MLP), redes neuronales convolucionales (CNN), redes neuronales recurrentes (RNN).

2.4. Redes Neuronales Artificiales

Una red neuronal artificial (ANN, por sus siglas en inglés) es un modelo computacional que trata de imitar el funcionamiento del cerebro humano, partiendo del elemento más básico que es la neurona. Estos modelos pertenecen al campo del aprendizaje de máquina (ML, por sus siglas en inglés) el cual se caracteriza por desarrollar algoritmos para crear una simulación del aprendizaje con el propósito de que una aplicación pueda adaptarse a condiciones inciertas o inesperadas [9]. Además las ANNs se diferencian de otros algoritmos del ML por aprender características de manera automática y representarlas jerárquicamente.

Las ANNs son una herramienta muy útil, su uso se ha popularizado en los últimos años debido a su versatilidad, poder y escalabilidad para cumplir con tareas extensas

y complejas, algunos ejemplos de estas tareas son: clasificación de imágenes, reconocimiento de voz, detección de tumores cerebrales, resumir grandes documentos automáticamente, realizar pronósticos de series de tiempo, etcétera [10].

Existen diferentes tipos de ANNs, las cuales se especializan en diferentes tipos de tareas, pero en general pueden clasificarse según el tipo de aprendizaje en:

- Aprendizaje Supervisado: Tipo de aprendizaje que se caracteriza por incluir en la información de entrenamiento la salida deseada, mejor conocida como etiquetas. El agente modificará iterativamente los parámetros de la red para obtener una mejor aproximación a las etiquetas. Una vez que el entrenamiento termina el agente es capaz de resolver tareas del entorno [11]. Las tareas más típicas que se resuelven con este tipo de aprendizaje son la clasificación y la regresión. Algunos de los modelos de aprendizaje supervisado son: Perceptrón Multicapa (MLP), Redes Neuronales Convolucionales (CNN) y Redes Neuronales Recurrentes (RNN).
- Aprendizaje No Supervisado: En este caso la información de entrenamiento no se encuentra etiquetada, por lo que el agente tiene que aprender características importantes para descubrir relaciones desconocidas. Las tareas más comunes del aprendizaje no supervisado son el clustering (agrupamiento) y la reducción de dimensionalidad. Algunos modelos son: Auto-Encoders (AE) y Restricted Boltzmann Machines (RBM).
- Aprendizaje Por Refuerzo: Este tipo de aprendizaje es utilizado cuando no se conoce el entorno. El agente observa su entorno, con base a ello selecciona y ejecuta una acción, la cual dará como resultado una recompensa o una penalización. El agente debe aprender automáticamente la mejor estrategia (política) que garantice la mayor recompensa [10].

2.4.1. Neurona biológica

El sistema nervioso humano está conformado por una compleja red de millones de neuronas interconectadas cuya función es transmitir información a través de impulsos nerviosos de un lugar del cuerpo a otro. Se estima que el ser humano tiene 10^{11} neuronas, cada una conectada en promedio con otras 10^4 [12].

Una de sus principales tareas por la cual es inspiración para las ANNs, es llevar a cabo el proceso de aprendizaje, en donde se consigue conocimiento al reforzar ciertas conexiones a través de experiencias obtenidas (entrenamiento).

La neurona biológica cuya anatomía se ilustra en la Figura 2.3, está conformada por una estructura principal que es el cuerpo de la célula dentro del cual se aloja el núcleo. A lo largo del cuerpo surgen diversas ramificaciones llamadas dendritas, además de una fibra más larga, conocida como axón. El axón puede ser hasta miles de veces más largo

que el cuerpo de la célula y a la vez este se divide en ramas llamadas telodendritas, al final de estas ramificaciones existen unas pequeñas estructuras llamadas terminales sinápticas las cuales se conectan con las dendritas de otras neuronas.

La sinapsis es el proceso mediante el cual se comunican las neuronas, este se lleva a cabo después de que una descarga química genera un impulso eléctrico a través de la neurona y recorre el axón desde el núcleo hasta las terminales sinápticas en donde la neurona segrega compuestos químicos (neurotransmisores) que son recibidas por las dendritas de otra neurona; Si la neurona post sináptica recibe suficiente cantidad de estos neurotransmisores dentro de los siguientes milisegundos, ésta dispara su propio impulso eléctrico, lo que quiere decir que la neurona esta excitada, aunque también existen neurotransmisores que inhiben la activación de la misma.

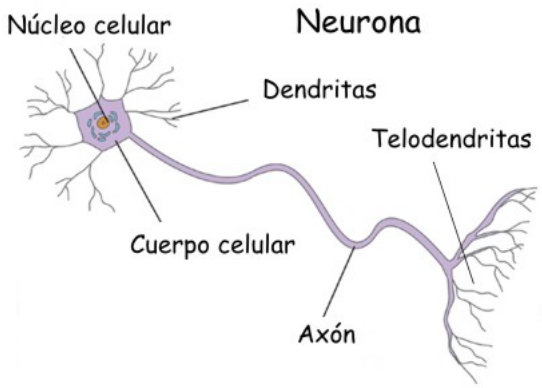

Figura 2.3: Neurona biológica. Recuperado de `http://www.iessuel.es/ccnn/interactiv/s_nervioso/s_nervioso_y_endocrino_06.htm`

2.4.2. Perceptrón

A partir de diferentes modelos de neuronas artificiales Frank Rosenblatt propuso en 1957 el Perceptrón, una arquitectura de ANNs simple que consiste básicamente en nodos (neuronas) con entradas y salidas representadas con números, donde cada entrada tiene un peso asociado. La Figura 2.4 ilustra como se encuentra conectado el perceptrón.

Las neuronas se entrenan al actualizar los pesos asociados para obtener la salida deseada. La actualización se hace a través de una suma ponderada de todas las entradas ($z = w_1x_1 + w_2x_2 + ... + w_nx_n = \mathbf{x}^T\mathbf{w}$) a la que después se le aplica una función escalón $h_w(\mathbf{x}) = step(z)$, donde $z = \mathbf{x}^T\mathbf{w}$.

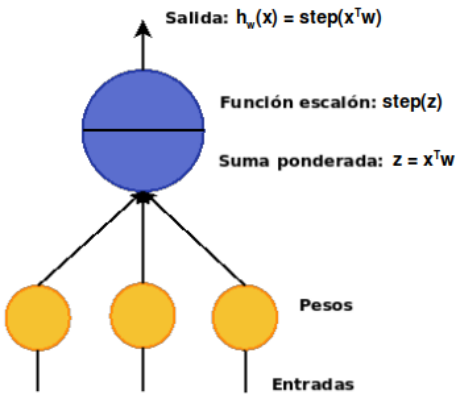

Figura 2.4: Neurona artificial.

2.4.3. Perceptrón Multicapa (MLP)

A pesar de que el perceptrón es un modelo muy efectivo para realizar clasificación, tiene muchas limitaciones, ya que no puede resolver problemas complejos (p. ej. compuerta lógica XOR) que no son linealmente separables.

Este problema puede solucionarse al conectar múltiples perceptrones. Este modelo es conocido como perceptrón multicapa (MLP, por sus siglas en inglés) y está conformado por múltiples capas de neuronas: la capa de entrada, una o más capas ocultas y una capa final llamada capa de salida. Todas las capas excepto la última tienen una neurona de bias. En la Figura 2.5 se puede ver como son las conexiones del MLP.

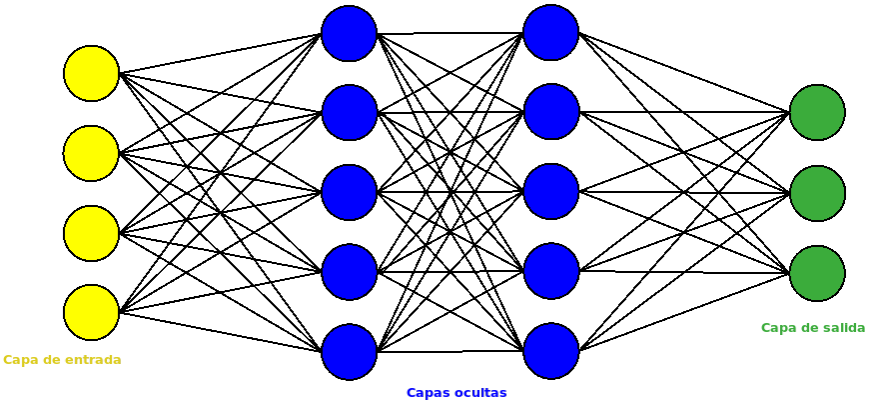

Figura 2.5: Arquitectura del perceptrón multicapa con cuatro neuronas de entrada, dos capas ocultas con cinco neuronas y tres neuronas de salida (las neuronas de bias están implícitas).

El algoritmo para entrenar los MLPs es conocido como backpropagation y fue desarrollado en 1986 por Rumelhart, Hinton y Williams. Con este algoritmo se consigue que un MLP autoajuste sus parámetros para aprender una representación interna de la información que se procesa.

Este método permite obtener las derivadas parciales (gradientes) de la función de costo con respecto a cada uno de los parámetros de la red, lo cual nos indica como varía dicha función de ante la variación de cada parámetro.

A continuación se muestra el algoritmo como se describe en M. Nielsen [13]:

1. Entradas x: Establecer la activación a^1 para la capa de entrada.
2. Propagación hacia adelante: Para cada capa $l = 2, 3, ..., L$ calcular $z^l = w^l a^{l-1} + b^l$ y $a^l = \sigma(z_l)$.
3. Error de salida δ^L: Calcular el vector $\delta^L = \nabla_a C \odot \sigma'(z^L)$.
4. Propagación hacia atrás del error: Para cada capa $l = L-1, L-2, ..., 2$ calcular $\delta^l = ((w^{l+1})^\intercal \delta^{l+1}) \odot \sigma'(z^l)$.
5. El gradiente de la función de costo está dado por $\frac{\partial C}{\partial w^l_{jk}} = a^{l-1}_k \delta^l_j$ y $\frac{\partial C}{\partial b^l_j} = \delta^l_j$.

Para finalizar, el algoritmo backpropagation realiza el descenso de gradiente con el cual se encuentran los mínimos locales para optimizar los parámetros de la red.

Otra cambio en la arquitectura de los MLPs, es utilizar funciones de activación que agreguen deformaciones no lineales a las salidas, esto con el propósito de que el

descenso de gradiente pueda obtener progresos en cada paso, ya que el algoritmo no puede moverse sobre superficies planas como es la función escalón [10].

A continuación se mencionan algunas de las funciones de activación que se utilizan en este trabajo cuyas gráficas se muestran en el Figura 2.6:

- Función sigmoide: $\sigma(z) = \frac{1}{(1+e^{-z})}$.
- Función tangente hiperbólica: $tanh(z) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$.
- Función de activación lineal rectificada (ReLU): $ReLU(z) = max(0, z)$.
- Leaky ReLU: $LeakyReLU(z) = max(\alpha z, z)$, donde α es una constante.

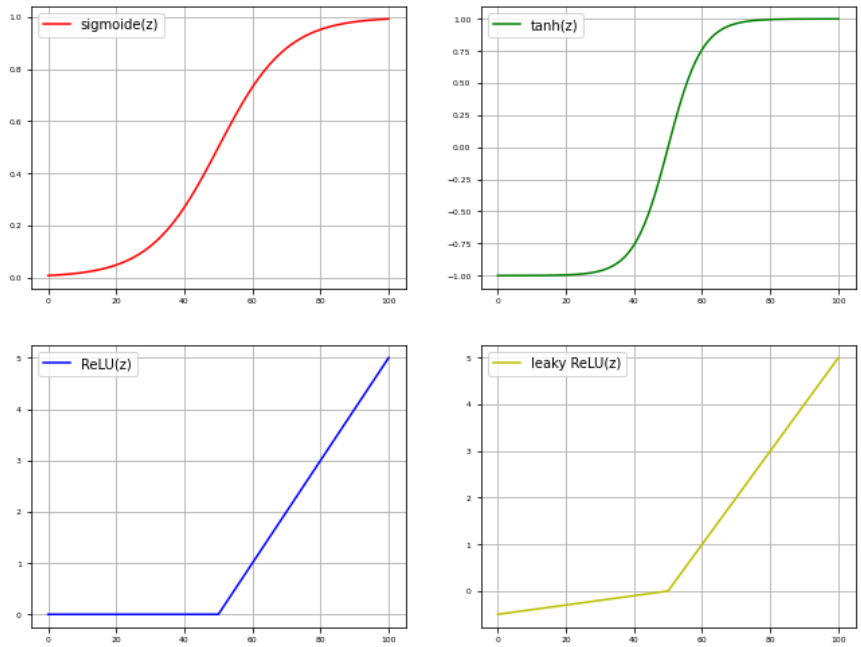

Figura 2.6: Funciones de activación.

2.4.4. Redes Neuronales Convolucionales (CNN)

Las redes neuronales convolucionales (CNN, por sus siglas en inglés) son una estructura de redes neuronales propuesto en 1988. Pero fue en los años 90s que trabajos como el de LeCun et al. [14] que agregan a las CNNs un algoritmo de aprendizaje basado en gradientes, que mejoraron su desempeño en problemas de clasificación de dígitos escritos a mano. Desde entonces este modelo continuó mejorando para realizar

tareas de reconocimiento. Las CNNs son utilizadas principalmente para el reconocimiento de imágenes y vídeo, sistemas de recomendación y procesamiento del lenguaje natural (NLP, por sus siglas en inglés). A diferencia de los MLPs, las CNNs tienen una arquitectura optimizada para procesar imágenes, tal y como lo hace la corteza visual, al extraer, reducir y aprender características clave para facilitar su clasificación. Por esta razón se utilizan principalmente para procesar imágenes (3-D o 2-D), pero no están limitadas a ello, ya que también se pueden utilizar con otro tipo de datos como series de tiempo o texto (1-D).

Como se mencionó anteriormente, su funcionamiento está inspirado en la organización de la corteza visual, en donde según un experimento realizado con gatos por David H. Hubel y Torsten Wiesel en 1959, existen neuronas especializadas para reaccionar cuando se exponen a líneas horizontales, y otras para hacerlo cuando se encuentran con diferentes orientaciones, en conjunto esta arquitectura es capaz de identificar patrones más complejos y conforma la percepción visual.

La arquitectura de las CNNs está conformada por una combinación de tres distintos tipos de capas: capa de convolución, capa de muestreo y capa de clasificación [11]. En donde se utilizan algunos elementos que ya se han descrito anteriormente como las capas completamente conectadas y las funciones de activación, pero también agregan otros elementos como la convolución y el muestreo. En la Figura 2.7 se puede ver como se encuentra configurada una CNN comúnmente.

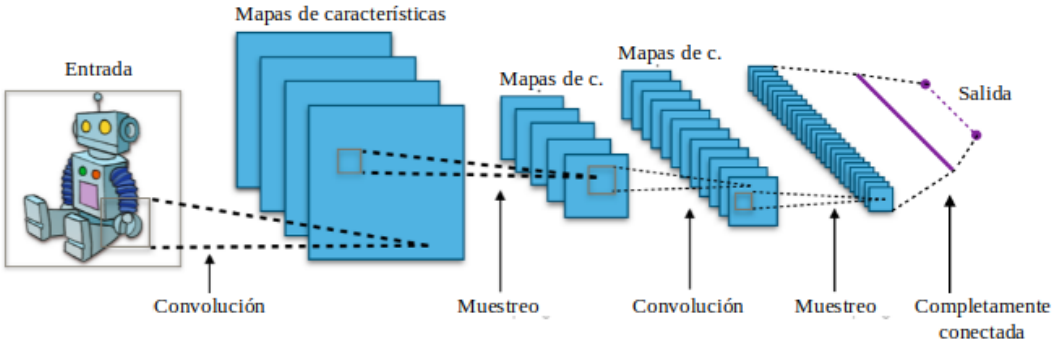

Figura 2.7: Arquitectura de una red convolucional (CNN), con una capa de entrada, alternancia entre múltiples capas de convolución y capas de muestreo y una capa totalmente conectada también llamada capa de clasificación. Recuperado de `https://en.wikipedia.org/wiki/Convolutional_neural_network#/media/File:Typical_cnn.png`

La **capa de muestreo** realiza un muestreo descendente sobre los mapas de características de entrada, seleccionando las neuronas más representativas. De esta manera se logra reducir el tamaño de la próxima capa de neuronas al conservar las características

más importantes de cada filtro. Esta operación se puede representar como

$$x_j^l = down(x_j^{l-1}), \tag{2.5}$$

donde down(.) representa una función de muestreo. Una de las funciones de muestreo más comunes es max-pooling, en donde se preserva el valor más alto de cada parche $N \times N$ de cada mapa de características, reduciendo el mapa de salida N veces.

La **capa de convolución**[1], se encarga de la aplicación de filtros a una imagen (matriz) para extraer patrones relevantes. Estos filtros son aplicados por núcleos que pueden aprender. La salida de cada núcleo pasa a través de una función de activación. A cada una de estas salidas se les conoce como mapa de características. Esta capa se puede representar como

$$x_j^l = f(\sum_{i \in M_j} x_i^{l-1} * k_{ij}^l + b_j^l), \tag{2.6}$$

donde x_j^l es la salida de la capa actual, x_i^{l-1} la salida de la capa anterior, k_{ij}^l es el núcleo para la capa actual y b_j^l son los biases para la capa actual. M_j representa los mapas de entrada.

En la Figura 2.8 se muestra un ejemplo de la convolución en 2 dimensiones para tareas de visión por computadora.

[1]Una convolución es una operación matemática que desliza una función sobre otra, resultando en una nueva.

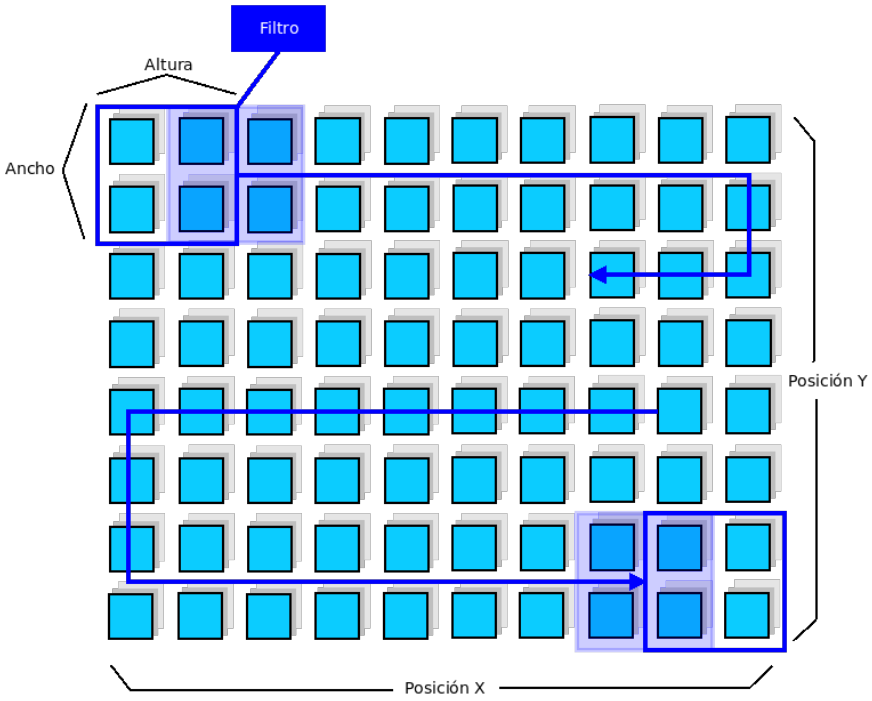

Figura 2.8: Ejemplo de convolución 2D para visión por computadora, cada pixel dentro de la imagen está dado por una posición X,Y y también tiene tres valores (RGB). El filtro se desplaza sobre la imagen tanto horizontal como verticalmente.

La **capa de clasificación** consta de una capa completamente conectada y se encarga de tomar las características extraídas por las capas anteriores para calcular su puntuación y determinar la clase a la que pertenecen.

2.4.4.1. Redes Convolucionales de 1 Dimensión (1D-CNN)

Ya que una de las principales tareas que atienden las CNNs es la clasificación de imágenes, este tipo de redes son un elemento importante de la visión por computadora, y por lo tanto, es bastante común que los núcleos de convolución sean de 2 dimensiones. Sin embargo, las redes convolucionales de 1 dimensión (1D-CNN) son útiles para obtener características relevantes de segmentos más cortos del conjunto de datos original y donde su ubicación no es muy relevante. Esto es útil para el análisis de series de tiempo o procesamiento del lenguaje natural (NLP, por sus siglas en inglés).

Las CNNs, ya sean de 1-D, 2-D o 3-D siguen el mismo principio, en donde la única diferencia es la dimensionalidad de los datos de entrada y como se desplazan los filtros a través de los datos. La Figura 2.9 se muestra un ejemplo de convolución 1-D NLP.

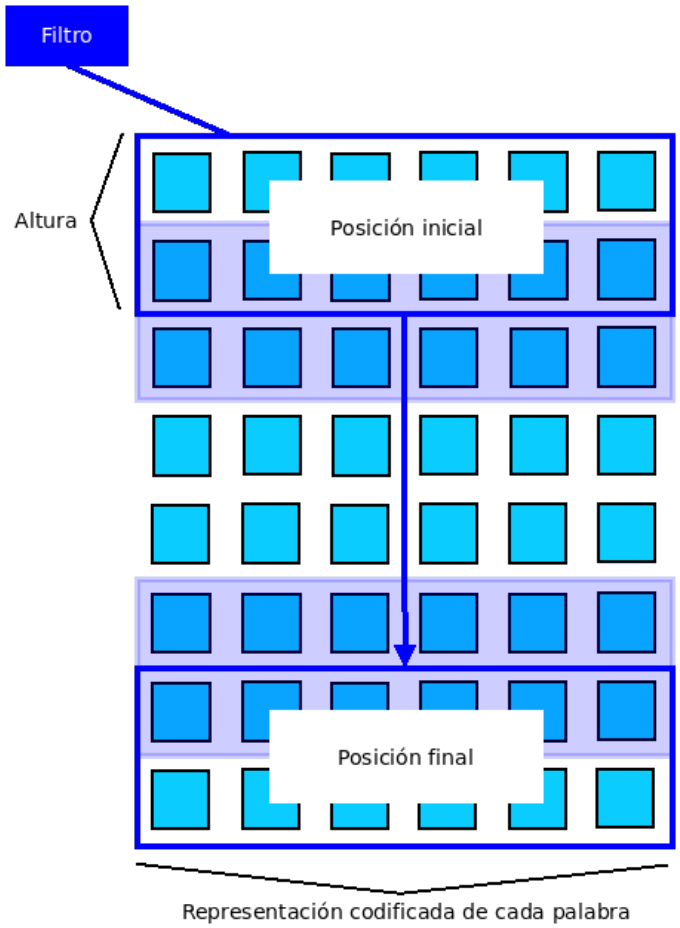

Figura 2.9: Ejemplo de convolución 1-D para NLP con una sentencia compuesta por 8 palabras. Cada palabra tiene una representación codificada. El filtro cubre la palabra completa. La altura determina cuantas palabras son consideradas para el entrenamiento del filtro.

2.4.4.2. Redes Convolucionales Profundas de Múltiples Canales (MC-DCNN)

Las redes convolucionales profundas de múltiples canales (MC-DCNN, por sus siglas en inglés) [15] son un modelo desarrollado para la clasificación de series de tiempo multivariadas. El modelo extrae y aprende las características de cada una de las variables que conforman una MTS individualmente y combina esta información para poder ser clasificada.

Tal y como su nombre lo indica, las MC-DCNNs se caracterizan por tener múltiples canales para la información de entrada, cada canal cuenta con una red convolucional

compuesta por sus respectivas capas de convolución y de muestreo. En contraste con otras CNNs que tratan con imágenes, en donde las convoluciones son en 2 dimensiones, para MTS, las MC-DCNN utilizan convoluciones de 1 dimensión en cada canal. Las salidas de cada canal se concatenan en una representación de características, de esta manera una capa conformada por un MLP que aprende las relaciones que hay entre las variables y se encarga de hacer la clasificación. En la Figura 2.10 se ilustra un ejemplo de la arquitectura de una MC-DCNN.

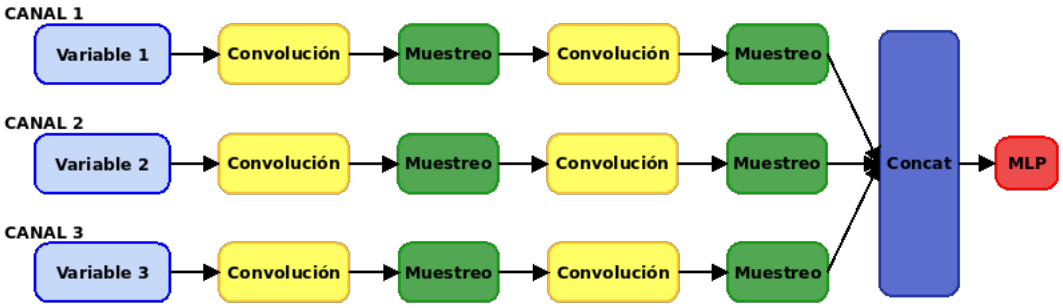

Figura 2.10: Arquitectura de una MC-DCNN. Conformada por tres canales de entrada, dos capas de convolución, dos capas de muestreo y una capa totalmente conectada (MLP).

2.4.5. Redes Neuronales Recurrentes (RNN)

A pesar de que los MLPs y CNNs son muy útiles para resolver una gran cantidad de problemas, existen situaciones con las que estos enfoques no responden favorablemente. Este tipo de problemas se caracterizan porque necesitan que haya persistencia de información del pasado, que permita descubrir correlaciones temporales que haya en todos los eventos de entrada, para hacer una clasificación de manera correcta. Lo que sería parecido a como los humanos recuerdan lo que ha pasado en un corto periodo de tiempo.

Un ejemplo claro de esto es la traducción de textos, en donde para comprender la oración no solo se tiene que analizar el significado de cada palabra tal y como lo haría un MLP, sino también, es necesario mantener el significado de las palabras anteriores las cuales pueden dar un contexto diferente a la palabra actual.

Las redes neuronales recurrentes (RNN, por sus siglas en inglés) tratan con datos secuenciales, implementando bucles que permiten la persistencia de la información. Las RNN están conformadas por un bloques de MLP idénticas, una para cada momento (t). En la Figura 2.11 se muestra como cada bloque de memoria, recibe una entrada externa y un estado anterior, produciendo una salida actual y un estado actual.

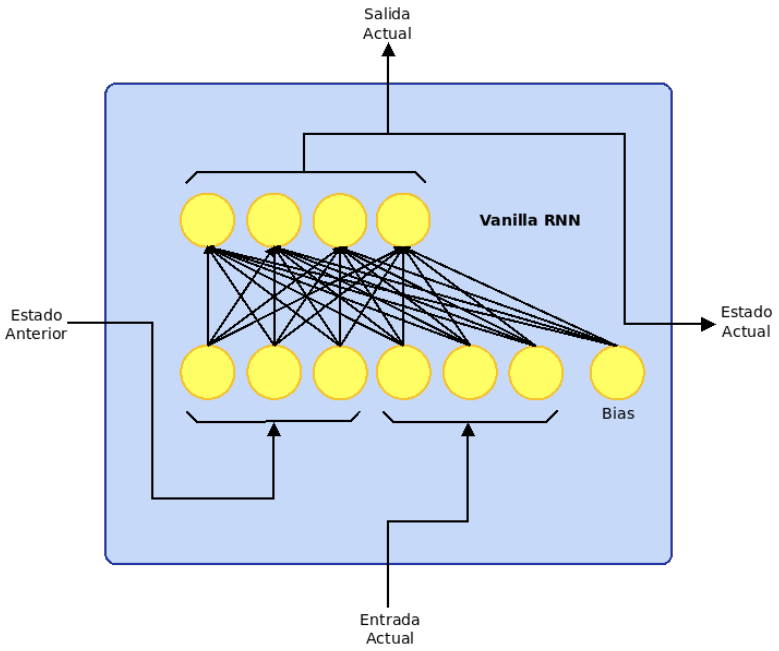

Figura 2.11: Estructura interna de una celda de una RNN básica, compuesta por una red neuronal de una sola capa cuya salida es la misma para el estado actual y la salida actual.

Este tipo de redes sigue el mismo principio que los modelos anteriores para el entrenamiento, salvo algunas modificaciones. En el caso del MLP, el entrenamiento se realiza mediante el algoritmo de Backpropagation, encargado de propagar el error hacia todas las capas de la red para obtener el vector gradiente y optimizarlo mediante el descenso de gradiente. Por otra parte en las RNNs no solo se tiene propagar el error a todas las capas de la red, sino que también es necesario propagarlo en cada instante de tiempo t hasta t=0, para posteriormente utilizar el descenso de gradiente. A este algoritmo se le conoce como **propagación hacia atrás a través del tiempo (BPTT, por sus siglas en inglés)**.

Sin embargo, las RNN tienen algunas dificultades a la hora de efectuar el entrenamiento. Uno de estos problemas surge cuando al momento de calcular el gradiente, los valores de los pesos recurrentes son muy pequeños, ya que esto ocasiona que el valor del gradiente disminuya en cada iteración, dicha disminución será cada vez más significativa al propagar el error hacia los demás tiempos, lo que resulta en el **desvanecimiento del gradiente**. El desvanecimiento del gradiente provoca que la red sea incapaz de aprender la correlación entre eventos temporalmente distantes. Por otra parte si el valor de los pesos recurrentes es muy grande el valor del gradiente crecerá en gran medida durante el entrenamiento, provocando lo que se conoce como explosión del gradiente que evita que la red converja [16].

2.4.5.1. Memoria a Corto y Largo Plazo (LSTM)

La memoria a corto y largo plazo (LSTM, por sus siglas en inglés) es un modelo propuesto en 1997 [17] que mejora las RNNs. Este modelo fue diseñado para evitar los problemas que se presentan durante el entrenamiento descritos anteriormente, dotándolas con la capacidad de recordar información por un largo periodo de tiempo. Desde que aparecieron se han propuesto diferentes enfoques basados en LTSM como los que se muestran en [11].

Las LSTM toman la estructura general de una RNN básica, pero cambia la estructura interna de cada celda de memoria como se muestra en la Figura 2.12.

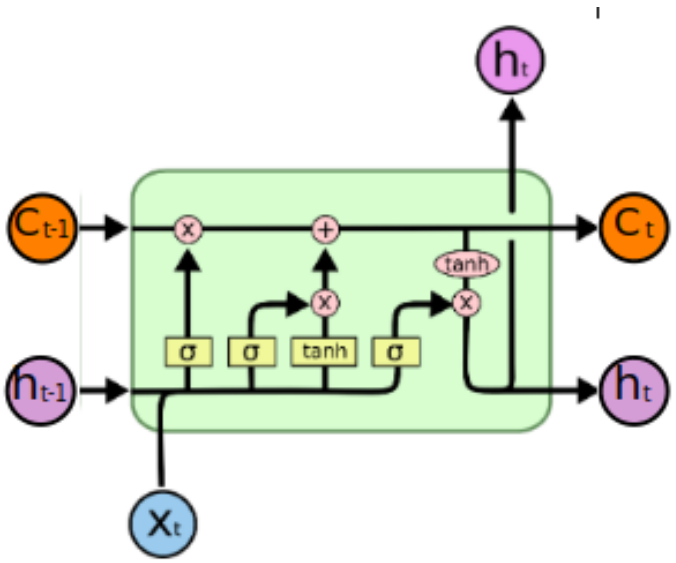

Figura 2.12: Estructura interna de una celda LTSM. Recuperado de `https://colah.github.io/posts/2015-08-Understanding-LSTMs/`

El objetivo de esta estructura es controlar el flujo de información en la celda de estado. La celda de estado (C) es una memoria selectiva que permite al modelo acceder a información de largos periodos de tiempo. Para remover o agregar información a la celda de estado, se utilizan unas estructuras llamadas compuertas, las cuales indican la cantidad de información que pasa a través de ellas. Las compuertas están conformadas por una función sigmoide de la red neuronal y la multiplicación punto a punto. El modelo LSTM tiene tres compuertas: compuerta de olvido (f_t), compuerta de entrada (i_t) y compuerta de salida (o_t).

La compuerta de olvido (f_t) se encarga de seleccionar qué información debe persistir en la celda de estado. Esta capa toma el estado anterior (h_{t-1}) y la entrada actual (x_t).

La compuerta de entrada (i_t) decide qué nueva información se debe de agregar en la celda de estado, pasando así de la celda de estado anterior (C_{t-1}) a la celda de estado actual (C_t). Por último se debe de decidir qué se debe mostrar a la salida, esta salida está basada en la celda de estado, que es filtrada por la compuerta de salida (o_t). Lo anterior puede definirse como:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{2.7}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{2.8}$$

$$\tilde{C}_t = tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{2.9}$$

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{2.10}$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{2.11}$$

$$h_t = o_t * tanh(C_t) \tag{2.12}$$

2.4.5.2. Unidad Recurrente Cerrada (GRU)

La unidad recurrente cerrada (GRU) es una variante de la LSTM introducida en 2014 por Cho, et al. [18]. Algunas de las diferencias de las GRUs con respecto a las LSTM son que las GRUs combinan las compuertas de olvido y de entrada en una sola compuerta llamada compuerta de actualización y combinan la celda de estado con el estado oculto. La GRU se expresa con las siguientes ecuaciones:

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t]) \tag{2.13}$$

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t]) \tag{2.14}$$

$$\tilde{h}_t = tanh(W \cdot [r_t * h_{t-1}, x_t]) \tag{2.15}$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t \tag{2.16}$$

2.4.5.3. Redes Neuronales Recurrentes Bidireccionales (BRNN)

Las redes neuronales recurrentes bidireccionales (BRNN, por sus siglas en inglés) fueron introducidas en 1997 [19]. El objetivo principal es incrementar la cantidad de información de entrada disponible, al entrenar de manera simultanea en dirección del tiempo positiva y negativa. Para lograrlo se duplica la capa oculta recurrente con dirección opuesta y se conectan a la misma salida. A la primera capa le es proporcionada la información de entrada, mientras que a la segunda se le proporciona una copia invertida de la información de entrada.

Este principio se ha aplicado con resultados favorables con LSTM y GRU principalmente en el campo del reconocimiento de voz como puede verse en las referencias [20] y [21].

2.4.6. Redes Generativas Antagónicas (GAN)

Las redes generativas antagónicas (GAN) son un modelo generativo de aprendizaje automático desarrollado por Goodfellow et al. en 2014 [22]. El modelo es capaz de crear aleatoriamente muestras de datos (imágenes, texto, sonido, etc.) nuevas muy similares a los datos de entrenamiento.

Las GANs estás conformadas por dos redes neuronales que se enfrentan entre sí en un juego de suma cero, en donde la pérdida de una de las redes se compensa con la de la red opuesta. En la Figura 2.13 se muestra un diagrama conceptual de cómo está conformada una GAN.

A la primera red se le conoce como **generador (G)** y se encarga de crear muestras de aquello que queremos generar a partir de un vector de entrada aleatorio que se selecciona de un espacio latente predefinido (p. ej. distribución normal).

La segunda red es llamada **discriminador (D)** y se encarga de comparar la distribución de las muestras producidas por la red generadora (muestras sintéticas) con la de las muestras del conjunto original.

Durante el entrenamiento, el discriminador y el generador tienen dos objetivos distintos: el discriminador intenta identificar datos falsos de datos reales, mientras que el generador intenta aumentar el error del discriminador, es decir, "engañarlo" presentándole muestras sintéticas que parecen venir de la distribución de datos originales. Tal y como ejemplifica Goodfellow [23]: "Podemos pensar en el generador como un falsificador, que intenta hacer dinero falso, y el discriminador como un policía, tratando de permitir dinero legítimo y atrapar dinero falso". Ambas redes se encuentran en un juego minimax con la función V(D, G), que de acuerdo con [22] se expresa como:

$$min_G max_D V(D,G) = \mathbb{E}_{xP_{data}(x)}[log(D(x))] + \mathbb{E}_{zP_{data}(z)}[log(1 - D(G(z)))] \quad (2.17)$$

Al inicio del entrenamiento el generador hará muestras fallidas, cuya distribución no será nada cercana a la de los datos de entrenamiento. De igual forma el discriminador no será capaz de clasificarlas de manera correcta. A medida que transcurren las iteraciones de entrenamiento, el discriminador aprende del conjunto de entrenamiento aquellas características que debe tomar en cuenta para clasificar las muestras proporcionadas por el generador. Por otra parte el generador aprende a crear muestras que cumplan con las características necesarias para "engañar" al discriminador, obteniendo mejores resultados en cada iteración, hasta que la salida del generador es muy cercana a las muestras originales del conjunto de entrenamiento. Un punto muy importante, es que el generador no ve en ningún momento las imágenes originales, este aprende su distribución de los gradientes que se calculan durante el entrenamiento.

La versión más sencilla es conocida como Vanilla GAN, y se caracteriza porque tanto discriminador como generador están conformados únicamente por MLPs.

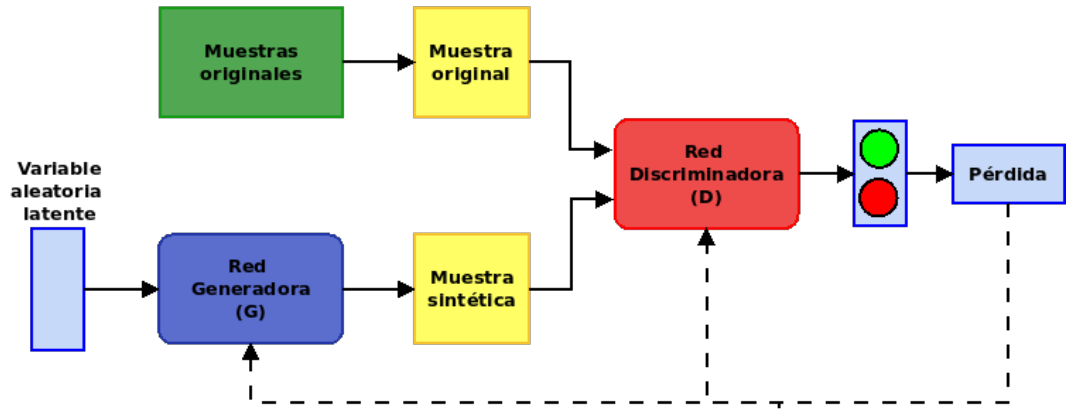

Figura 2.13: Diagrama conceptual de una Red Generativa Antagónica (GAN).

Una de las aplicaciones más populares de las GANs es la generación de imágenes fotorrealistas, lo cual se ha encontrado útil en diferentes aplicaciones de diseño [11]. Además del procesamiento de imágenes y vídeo, las GANs también hacen buen trabajo en el procesamiento de audio y el habla. Por ejemplo la llamada red generativa antagónica para mejora del habla (SEGAN, por sus siglas en inglés) que configura su diseño centrado en el habla y mejora su rendimiento [24]. Pero las GANs no están limitadas al campo multimedia, ya que también se han encontrado otro tipo de aplicaciones como el procesamiento de información médica, en donde existen modelos como las redes generativas antagonistas condicionales (cGAN, por sus siglas en inglés) capaces de segmentar tumores cerebrales [25].

2.4.6.1. Redes Generativas Antagónicas Convolucionales Profundas (DC-GAN)

Las redes generativas antagónicas convolucionales profundas (DC-GAN, por sus siglas en inglés) de 2015 [26] son un modelo mejorado de las GANs. A diferencia de Vanilla GAN o una GAN con redes convolucionales, las cuales solo pueden generar imágenes pequeñas, las DC-GANs tienen una serie de cambios, como la utilización de redes convolucionales profundas (DCNN, por sus siglas en inglés) que le permiten tratar bases de datos muy grandes. Esta mejora se consiguió después de experimentar con diferentes arquitecturas e hiper-parámetros. A continuación se mencionan algunos de estos cambios importantes para construir una DC-GAN:

- Remplaza las capas de muestreo (pooling) con convoluciones escalonadas en el discriminador, y convoluciones transpuestas (también llamadas deconvoluciones) en el generador.
- Utiliza Batch Normalization en ambos modelos, excepto en la capa de salida del generador y en la primera capa del discriminador.
- Utiliza la función de activación ReLU en el generador, excepto la capa de salida, la cual utiliza tanh.
- Utiliza la función de activación leaky ReLU en todas las capas del discriminador.

Ya que las DC-GAN fueron diseñadas para tratar con imágenes, utilizan capas de convolución con filtros 2-D.

2.4.6.2. Redes Generativas Antagónicas para Series de Tiempo Multivariadas (MTS-GAN)

Con el propósito de tratar con los datos faltantes en las MTS, se diseñaron las redes generativas antagónicas para series de tiempo multivariadas (MTS-GAN, por sus siglas en inglés) [2]. El cual aprovecha la capacidad del las GANs para modelar la distribución de los datos.

La MTS-GAN toma como base el modelo DC-GAN [26], y lo modifica para tratar con las limitaciones que tiene al modelar la distribución de las MTS. Una de los principales cambios que tiene esta versión, es que introduce el modelo MC-DCNN [15] dentro de la DC-GAN.

En red discriminadora, las características de cada variable de la MTS son extraídas por convoluciones, y luego las relaciones entre cada canal son modeladas por un MLP cuya salida es un escalar. Las capas de activación son las mismas que se utilizaron en

DC-GAN. La Figura 2.14 ilustra como está conformada la red discriminadora.

La red generadora se muestra en la Figura 2.15 es inversa a la red discriminadora. Es decir, parte de un vector aleatorio latente con distribución uniforme, y realiza convoluciones transpuestas (deconvoluciones) dentro de cada canal. Al final se obtendrá en la salida de cada canal una variable de la MTS. De igual forma las capas de activación son iguales a las de DC-GAN.

A diferencia de DC-GAN, todas las convoluciones y deconvoluciones en MTS-GAN se realizan aplicando filtros 1-D, tal y como se hace en MC-DCNN.

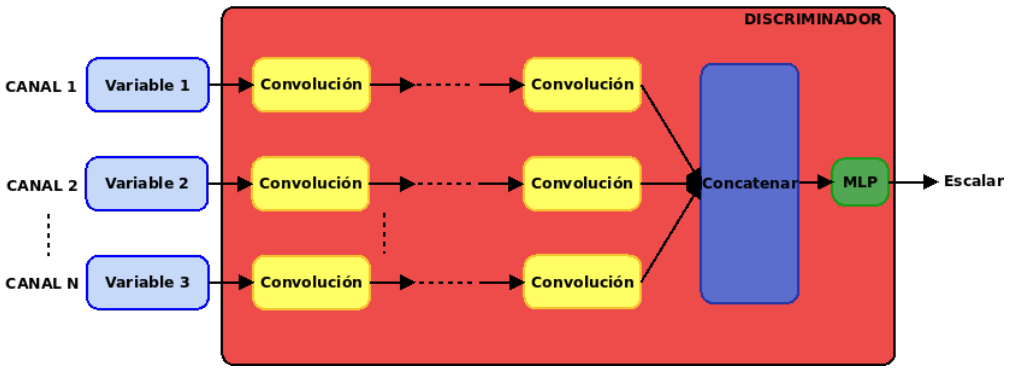

Figura 2.14: Discriminador de MTS-GAN para MTS de N dimensiones.

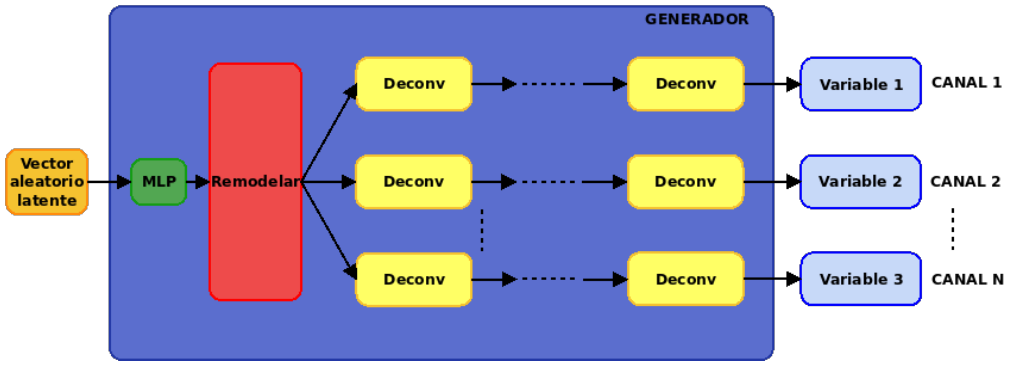

Figura 2.15: Generador de MTS-GAN para MTS de N dimensiones.

El algoritmo 1 muestra los pasos para realizar la imputación de datos faltantes en series de tiempo multivariadas.

Algorithm 1 Imputación de datos con MTS-GAN

Require: Conjunto de datos $\mathcal{X}$, factor de aprendizaje α y número de iteraciones back-propagation.

1: Dividir $\mathcal{X}$ en $\mathcal{X}^{com}$ y $\mathcal{X}^{inc} = \{X_m^{inc}\}_{m=1}^{M_2}$.
2: Tomar el vector latente aleatorio d-dimensional $Z \sim U([0,1]^d)$ con distribución uniforme como entrada, entrenar G y D de MTS-GAN con $\mathcal{X}^{com}$ para resolver el problema de optimización definido por (2.17).
3: Buscar $\mathcal{Z}$:
4: **for** k iteraciones **do**
5: Calcular la pérdida: $\mathcal{L}_{inc}(\mathbf{z}|\mathcal{X}^{inc}) = \sqrt{\frac{1}{M_2}\sum_{m=1}^{M_2} \frac{||(X_m^{inc} - G(\mathbf{z}_m)) \bigodot \delta_m^{inc}||_F^2}{sum(\delta_m^{inc})}}$
6: Actualizar $\mathbf{z}$ al descender sus gradientes: $\mathbf{z} \leftarrow \mathbf{z} - \alpha \nabla_{\mathbf{z}} \mathcal{L}_{inc}(\mathbf{z}|\mathcal{X}^{inc})$
7: Recortar $\mathbf{z}$ a $[0,1]$: $\mathbf{z} \leftarrow clip(\mathbf{z}, 0, 1)$
8: **end for**
9: Obtener $\widehat{\mathcal{Z}^{inc}} = \{\widehat{\mathbf{z}_m^{inc}}\}_{m=1}^{M_2}$ después de back-propagation e imputar $\{\mathbf{X}_m^{inc}\}_{m=1}^{M_2}$
10: **for** $m = 1$ a M_2 **do**
11: La muestra reconstruida se obtiene: $X_m^{rec} = X_m^{inc} \bigodot \delta_m^{inc} + G(\widehat{\mathbf{Z}_m^{inc}}) \bigodot (I - \delta_m^{inc})$
12: **end for**

Capítulo 3

Diseño del Experimento

La contaminación del aire sucede cuando algunas sustancias y partículas presentes en la atmósfera exceden ciertos niveles de concentración, modificando su composición natural. Este fenómeno ocurre en las capas más bajas de la atmósfera; la estratosfera y la troposfera, siendo esta última la capa en donde habitan los seres humanos y otros organismos [3]. Algunos de los contaminantes presentes en la atmósfera son: dióxido de azufre (SO_2), dióxido de nitrógeno (NO_2), dióxido de carbono (CO_2), monóxido de nitrógeno (NO), monóxido de carbono (CO), óxidos de nitrógeno (NO_X), partículas menores a 2.5 micrómetros ($PM_{2.5}$) y partículas menores a 10 micrómetros (PM_{10}). Estos contaminantes pueden ser generados tanto por fuentes naturales (p. ej. erupciones volcánicas) como por fuentes antropogénicas (p. ej. quema de combustibles) [8].

Las altas concentraciones de contaminantes del aire traen efectos perjudiciales para la salud de los seres vivos, particularmente en el caso de los seres humanos pueden ser enfermedades respiratorias, enfermedades cardiovasculares y cáncer, entre otros [3]. Además la contaminación del aire tiene un gran impacto sobre el medio ambiente, provocando problemas como el cambio climático y calentamiento global [1].

En la Zona Metropolitana del Valle de México (ZMVM) existen diferentes estaciones de monitoreo de calidad del aire distribuidas en toda la zona, estas estaciones se encargan de realizar mediciones hora tras hora de diferentes datos meteorológicos y de contaminantes del aire. El análisis de estos datos permite realizar predicciones sobre los niveles de concentración de los diferentes contaminantes del aire. Esta tarea es de gran importancia, ya que en caso de anticipar niveles altos de concentración de algunos contaminantes, ayuda a tomar las medidas necesarias para normalizarlas y resguardar la integridad de la población.

Como se ha mencionado en el Capítulo 2, los datos faltantes son un problema muy común en conjuntos de datos del mundo real y este caso no es la excepción. Los datos faltantes en este conjunto de datos, se debe principalmente a fallas mecánicas en los sensores encargados de realizar las mediciones. La cantidad de datos faltantes presen-

tes en cada estación de monitoreo varía bastante, teniendo casos en los que apenas se presenta un dato faltante por día, otros casos en los que se tienen meses, incluso años completos sin valores registrados.

En este capítulo se presenta información sobre las series de tiempo multivariadas (MTS) de contaminantes del aire, se describe la forma en que se obtiene el conjunto de datos, las soluciones propuestas en este trabajo para los datos faltantes y para el pronóstico de la serie de tiempo, se describen las acciones y pasos a seguir para su desarrollo como: la preparación de los datos, los modelos utilizados, entrenamiento, entre otras, detallando aquellos aspectos más importantes para la experimentación.

3.1. Conjunto de Datos

El Sistema de Monitoreo Atmosférico de la Cuidad de México (SIMAT) se encarga del monitoreo de la calidad del aire de la ZMVM. Se encuentra conformado por 4 subsistemas que se especializan en distintos aspectos.

Uno de estos subsistemas es la Red Automática de Calidad del Aire (RAMA), la cual cuenta con 34 estaciones de monitoreo (21 en CDMX y 13 en EDOMEX). Su función principal es medir y registrar los valores de concentración de contaminantes del aire como: monóxido de carbono (CO), monóxido de nitrógeno (NO), dióxido de nitrógeno (NO_2), óxidos de nitrógeno (NO_X), ozono (O_3), partículas menores de 10 micrómetros (PM_{10}), partículas menores de 2.5 micrómetros ($PM_{2.5}$), partículas gruesas (PMCO) y dióxido de azufre (SO_2). Otro subsistema es la Red de Meteorología y Radiación Solar (REDMET), compuesta por 16 estaciones (10 en CDMX y 6 en EDOMEX) encargadas de medir y registrar condiciones meteorológicas como: velocidad del viento, dirección del viento, temperatura y humedad relativa, presión atmosférica, radiación global, ración solar UV-A y UV-B. La Figura 3.1 muestra la ubicación de las distintas estaciones.

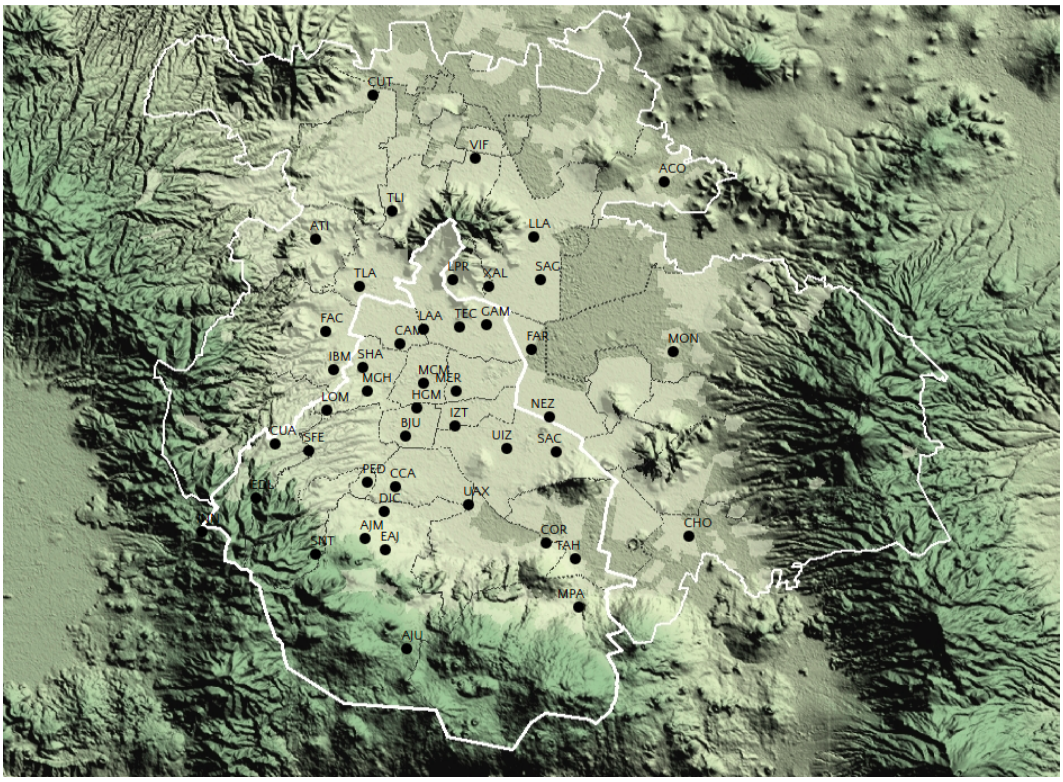

Figura 3.1: Ubicación de las estaciones de monitoreo de calidad del aire. Recuperado de `http://www.aire.cdmx.gob.mx/default.php?opc=%27ZaBhnmM=%27`

En algunos casos, las mediciones son registradas desde el año 1986, año en que inició operaciones RAMA con 20 estaciones que median CO, SO_2, NO_X y O_3. Además algunas de estas estaciones eran capaces de medir condiciones meteorológicas como temperatura, humedad relativa, dirección y velocidad del viento. Debido a esto, existen mediciones que son más nuevas y solo se encuentran disponibles a partir de años más recientes.

Los conjuntos de datos publicados por la SEDEMA[1] se encuentran en archivos de valores separados por comas (CSV, pos sus siglas en inglés), organizados por año, en donde, cada archivo contienen las mediciones de los todos los contaminantes mencionados en las estaciones disponibles. Las mediciones están registradas a intervalos de una hora, desde las 01:00 hrs. del 1ro de Enero hasta las 00:00 hrs. del 31 de Diciembre del año en cuestión. Por lo tanto, por cada contaminante en determinada estación deberá haber un total de 8,760 mediciones o 8,784 en el caso de los años bisiestos.

Para este trabajo se trabajó principalmente con la estación Merced (MER) debido

[1] `http://www.aire.cdmx.gob.mx/default.php?opc=%27aKBhnmI=%27&opcion=Zg==`

a que es una de las pocas estaciones que registra mediciones desde el año 1986, con un total de 298032 mediciones. Además contiene un mayor número de muestras completas, es decir, días completos sin la datos faltantes, con un total de 6383 días completos.

Las mediciones disponibles en la estación Merced fueron: O_3, CO, NO_2, NO_X, SO_2, velocidad del viento, dirección del viento, temperatura y humedad relativa. En Tabla 3.1 se muestran las estadísticas descriptivas del conjunto de datos de la estación Merced, mientras que la Figura 3.2, 3.3, 3.4, 3.5, 3.6, 3.7 y 3.8 sus gráficas.

	O_3	**CO**	**NO_2**	**NO_X**	**SO_2**	**Vel. viento**	**Dir. viento**	**Hum. Rel.**	**Temp.**
media	31.30	2.25	39.42	78.57	18.93	1.72	176.58	52.17	17.12
std	36.64	2.20	22.81	59.85	26.00	1.02	119.50	20.93	4.76
min	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.40
25 %	6.00	0.80	25.00	37.00	3.00	0.90	67.00	36.00	14.00
50 %	17.00	1.50	35.00	61.00	9.00	1.50	156.00	53.00	16.60
75 %	44.00	2.90	48.00	101.00	22.00	2.30	306.00	69.00	20.50
máx	380.00	31.30	344.00	537.00	406.00	8.80	360.00	100.00	34.40

Tabla 3.1: Estadísticas descriptivas del conjunto de datos de la estación Merced (MER).

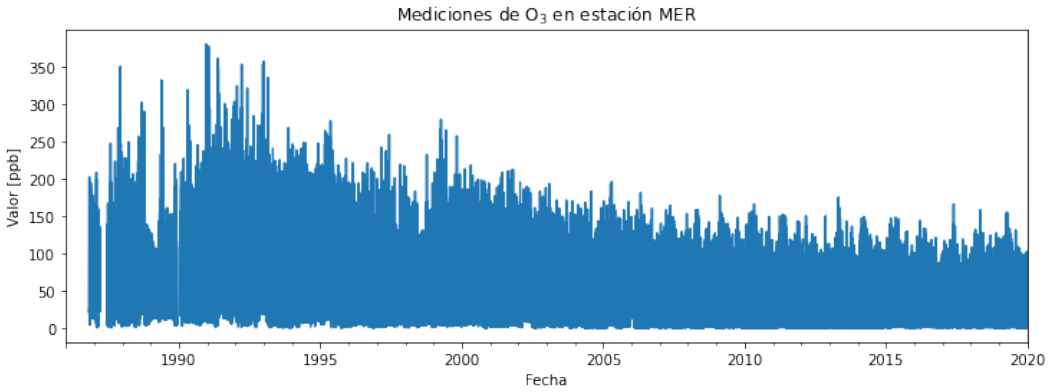

Figura 3.2: Mediciones de ozono en la estación Merced (MER).

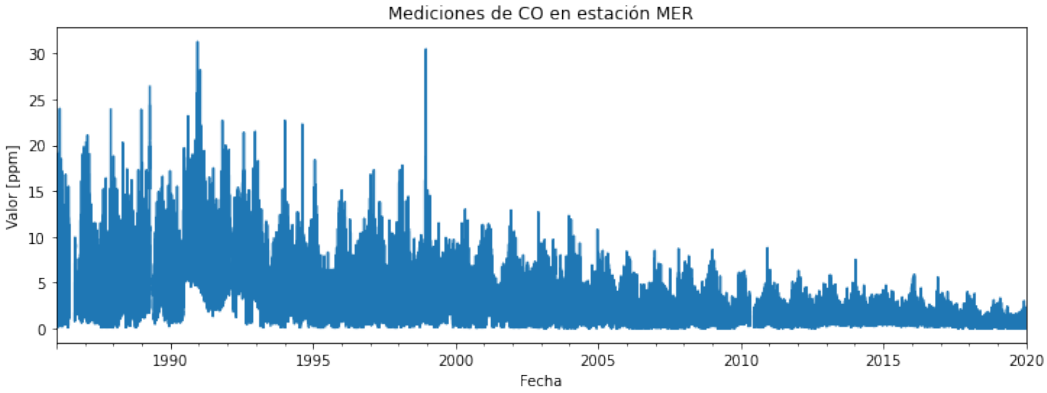

Figura 3.3: Mediciones de óxido de carbono en la estación Merced (MER).

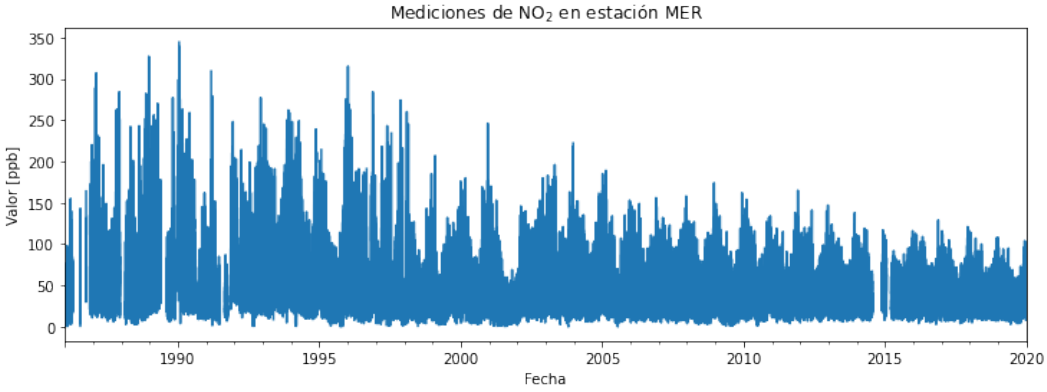

Figura 3.4: Mediciones de dióxido de nitrógeno en la estación Merced (MER).

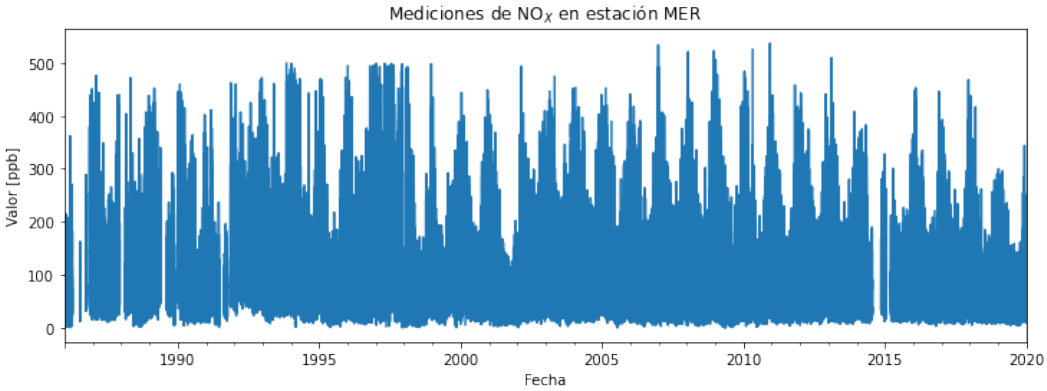

Figura 3.5: Mediciones de óxidos de nitrógeno en la estación Merced (MER).

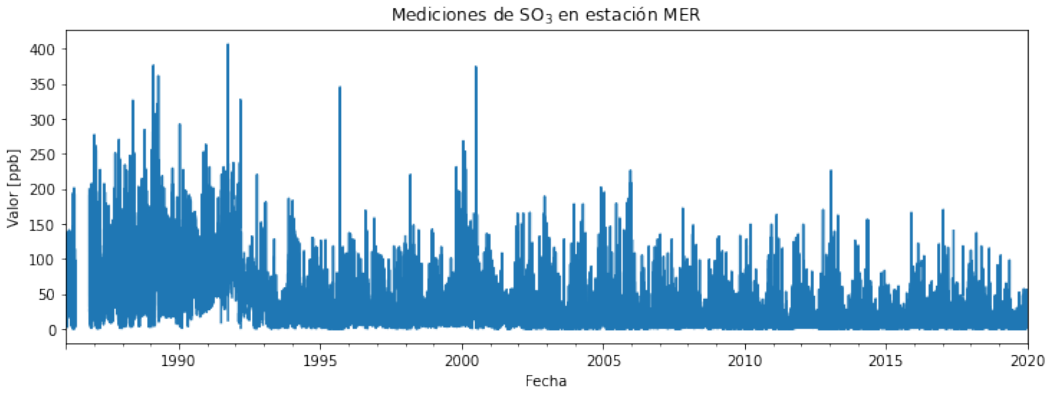

Figura 3.6: Mediciones de dióxido de azufre en la estación Merced (MER).

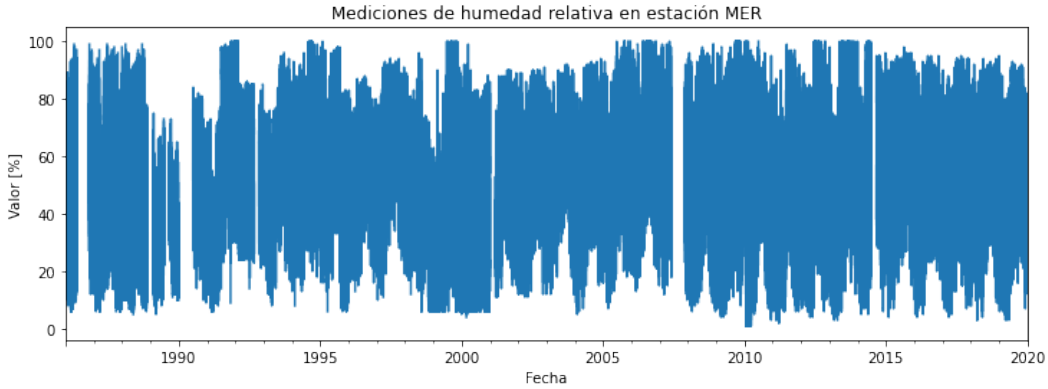

Figura 3.7: Mediciones de la humedad relativa en la estación Merced (MER).

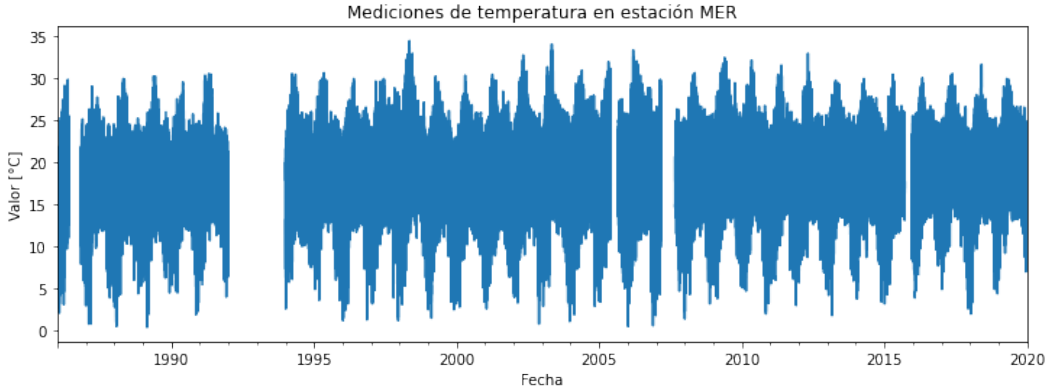

Figura 3.8: Mediciones de temperatura en la estación Merced (MER).

Debido a que los datos correspondientes a la dirección del viento están dados en grados, no se agrega una representación visual, pues esta no tendría mucho sentido con este formato. Más adelante se utiliza la dirección y velocidad del viento para construir características que permitan visualizar mejor estos datos, además que sean más adecuados para los modelos que se utilizarán.

Las gráficas son útiles para realizar un análisis de las series de tiempo, ya que muestran como es la distribución de cada uno de los contaminantes y condiciones meteorológicas. También ayudan a visualizar la tendencia, la estacionalidad, posibles valores atípicos y datos faltantes.

3.2. Preprocesamiento de datos

A partir del conjunto de datos anterior, se procedió a formar series de tiempo univariadas de cada medición tanto en la estación Merced como en las demás estaciones disponibles. Estas series se utilizarían posteriormente para formar series de tiempo multivariadas de tres variables, utilizando aquellas variables con mayor correlación entre ellas.

Se modificó la forma de las series de tiempo multivariadas para que fueran compatibles como entrada de los modelos que aquí se utilizan, la forma utilizada fue (muestra, longitud, variable). En donde, una muestra es equivalente a las mediciones de un día, ya que las mediciones son registradas cada hora, cada muestra tiene una longitud de 24 horas y la variable indica el número de características que se registran.

3.2.1. Extracción de características

Antes de proponer algún modelo, es importante que los datos sean comprendidos y que se extraiga de ellos toda información relevante para su procesamiento, ya que del conjunto de datos proporcionado se pueden obtener características nuevas que ayuden a encontrar nueva información útil sobre la serie de tiempo. Otro punto importante, es asegurarse de que los datos tengan el formato correcto y que sea fácil de interpretar para los modelos que se utilizarán.

3.2.1.1. Hora del día

De la fecha y hora de la serie de tiempo, se pueden obtener características muy importantes, las cuales pueden brindar información relevante sobre la estacionalidad de la serie [28]. Por lo tanto, los modelos que se utilicen posteriormente, tendrán acceso a las características importantes sobre el tiempo.

Sin embargo, los datos correspondientes al tiempo no pueden ser suministrados al modelo directamente. Como sabemos, atributos de tiempo como: años, meses, días, horas y segundos son cíclicos, por lo tanto, es muy importante codificar esta información de tal manera que el modelo sepa que estos atributos contienen esta característica.

En la Figura 3.9 se observa la señal de hora del día sin ser codificada a lo largo de 96 horas. En dicha figura claramente pueden observarse los saltos abruptos que ocurren cuando consideramos las 23:00 y 00:00, dando una diferencia de 23 horas cuando en realidad es de tan solo 1 hora. Este es el principal problema de proporcionarle datos cíclicos a un modelo de machine learning o deep learning.

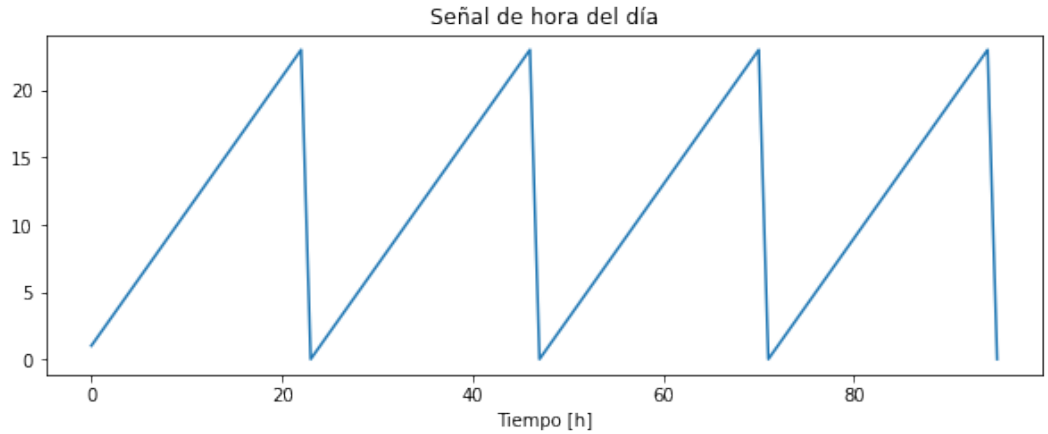

Figura 3.9: Señal de hora del día en la estación Merced (MER) sin codificación.

Existen diferentes métodos para codificar datos cíclicos, en este caso se optó por aplicar transformaciones de seno y coseno sobre las características a utilizar. Para codificar esta información se crearon dos nuevas características que indicarán la hora del día. Las nuevas características creadas fueron DSIN=$\sin(2\pi hora/24)$ y DCOS=$\cos(2\pi hora/24)$. La transformación coseno es igual de importante para que la codificación sea exitosa, ya que permite diferenciar los puntos en los que la transformación seno tiene los mismos valores. La Figura 3.10 muestra las señales seno y coseno de la hora del día en 24 horas. en donde podemos observar que ya no existen esos saltos abruptos cuando pasamos de 23:00 a 00:00.

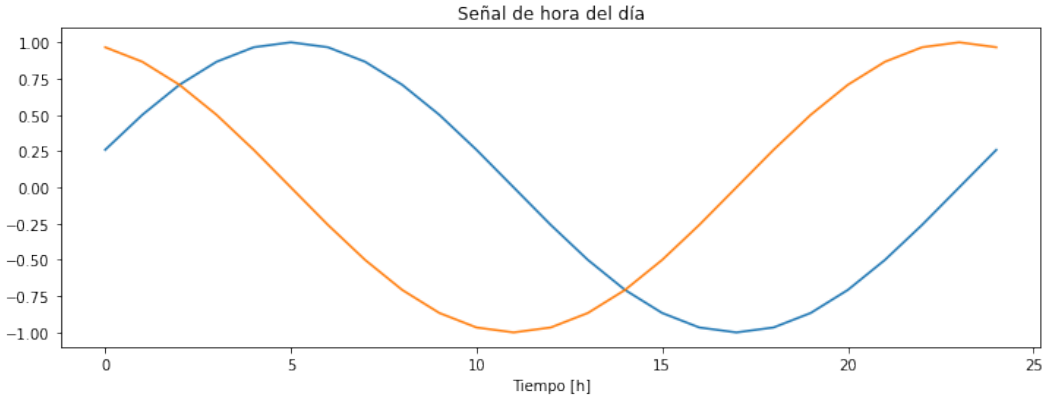

Figura 3.10: Señal de hora del día codificada en la estación Merced (MER).

En la Figura 3.11, se observa que ambas señales representan la información de manera cíclica, lista para ser procesada por nuestro modelo.

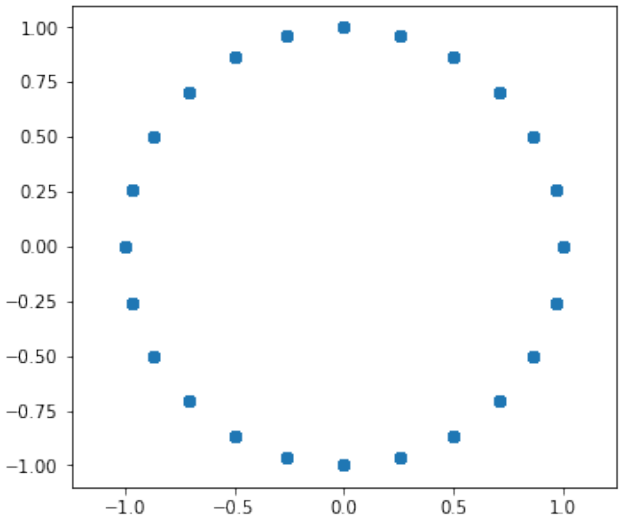

Figura 3.11: Gráfica en 2D de las señales seno y coseno.

3.2.1.2. Viento

Como se mencionó anteriormente, realizar una gráfica de dirección y velocidad del viento no es muy útil, pues no son buenas entradas para los modelos, esto se debe a que la dirección del viento está dada en grados y por lo tanto 0° y 360° son equivalentes. En su lugar, se utilizó esta información para construir un vector de viento, el cual representa el movimiento del aire sobre el suelo. Los vectores de viento brindan una distribución más sencilla para los modelos posteriores que se utilicen para realizar la

imputación de datos faltantes y el pronóstico de la serie de tiempo. La Figura 3.12 muestra la distribución de los vectores de viento.

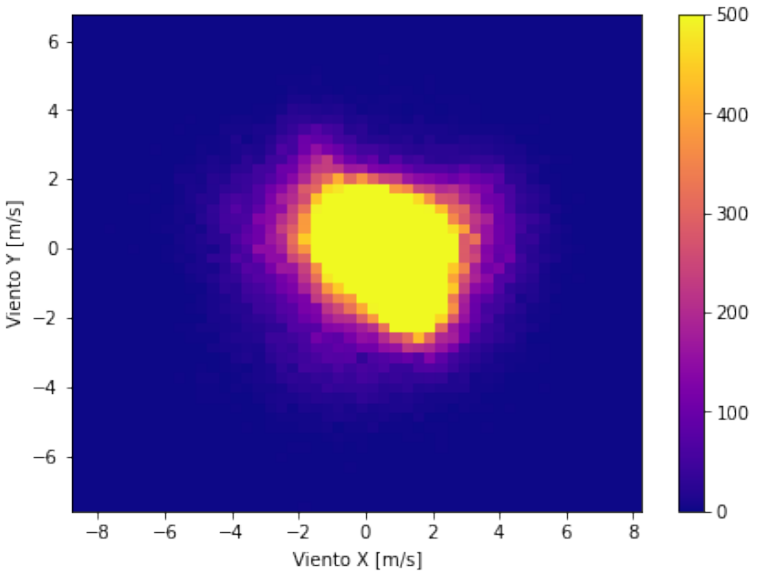

Figura 3.12: Vectores de viento en estación Merced (MER).

3.2.2. Correlación

Para seleccionar las variables que conformarían la serie de tiempo multivariada, se realizó una prueba para medir la dependencia lineal entre las variables, esta prueba es conocida como coeficiente de correlación de Pearson. La prueba ayuda a conocer si los valores de una variable tienden a aumentar (o disminuir) a medida que aumentan los valores de la otra. El valor que puede tomar este coeficiente se encuentra entre -1 y 1 y se interpreta de la siguiente manera:

- Si $r > 0$ existe una correlación positiva, ambas variables se relacionan directamente.
- Si $r < 0$ existe una correlación negativa, ambas variables se relacionan inversamente.
- Si $r = 1$ o $r = -1$ significa que existe una correlación lineal (positiva o negativa) perfecta entre ambas variables.
- Si $r = 0$ no existe relación lineal entre las variables.

Para visualizar mejor los coeficiente de correlación, se elaboró un mapa de calor, el cual indica los diferentes coeficientes de correlación mediante diferentes tonalidades de

una paleta de colores. La Figura 3.13 muestra dicho mapa, en donde los colores obscuros indican una correlación positiva, mientras que los colores claros indican correlación negativa.

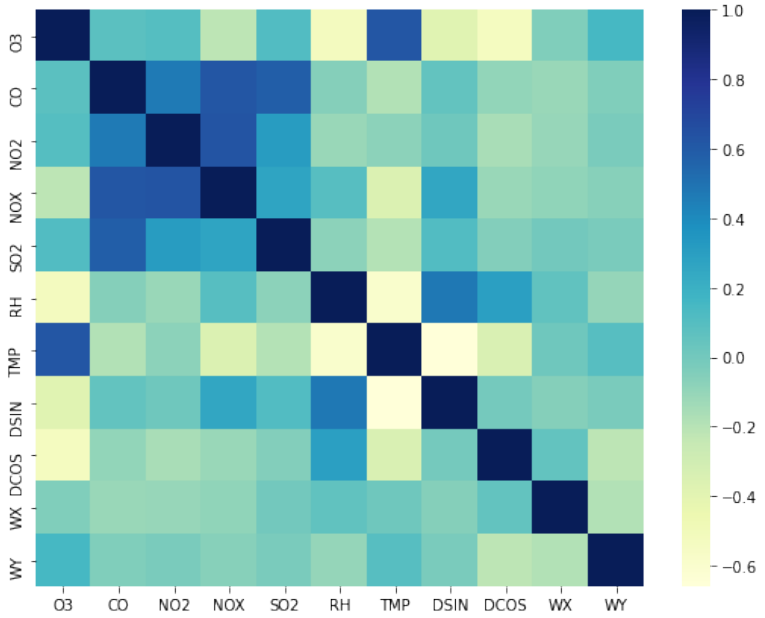

Figura 3.13: Mapa de calor de correlación.

Del mapa de calor anterior, se puede observar que se destacan características con respecto al O_3 como: NO_X (NOX), componente del viento horizontal (WX), componente del viento vertical (YX), temperatura (TMP) y humedad relativa (RH). Así que se tomarán estas variables como las más importantes debido a la correlación lineal que hay entre ellas. Para futuros proyectos, no se descarta la posibilidad de emplear otras de estas variables, o incluso, agregarlas todas al mismo modelo con el propósito de que el modelo aprenda de las correlaciones que existen entre todas ellas, sin embargo esta tarea exige más tiempo y recursos computacionales.

Tomando en cuenta el mapa anterior se construyeron series de tiempo multivariadas con las cuales se buscó la mejor opción para realizar un pronóstico principalmente de ozono. Las opciones fueron las siguientes:

- ozono, temperatura, humedad relativa: Serie conformada con 3 variables, las cuales guardan correlación entre ellas.

3.3. Experimentos

Una vez que se han construido las series de tiempo multivariadas sobre contaminantes del aire, la experimentación consistirá en evaluar y comparar los modelos propuestos para realizar dos tareas principales: la imputación de datos faltantes en series de tiempo multivariadas y los pronósticos de series de tiempo multivariadas.

3.3.1. Imputación de datos faltantes con MTS-GAN

Antes de realizar un pronóstico sobre la series de tiempo multivariadas creadas anteriormente (series de tiempo originales), es necesario tratar la gran cantidad de datos faltantes que presenta el conjunto de datos, para hacerlo, se propone el modelo de aprendizaje profundo MTS-GAN descrito en el Capítulo II.

La imputación se realizará sobre dos conjuntos creados a partir del las series de tiempo originales:

- Conjunto de datos de evaluación: Este conjunto se creará a partir de las series de tiempo originales. Los datos faltantes que se presenten en este conjunto serán insertados intencionalmente con el propósito únicamente de evaluar los métodos de imputación que se presentan.
- Conjunto de datos reales: Conjunto que estará formado por las series de tiempo originales tal y como fueron creadas. Los datos faltantes en este conjunto son auténticos. Al imputar los datos faltantes se obtendrá la serie de tiempo reconstruida sobre la cual se realizará el pronóstico.

3.3.1.1. Imputación sobre conjunto de datos de evaluación

Para ver que tan confiable es el modelo MTS-GAN frente algunos de los métodos más comunes para la imputación de datos, se realizará una comparación entre los resultados de estos métodos sobre un conjunto de datos de evaluación.

El conjunto de datos de evaluación se creará únicamente con las muestras completas de las series de tiempo multivariadas originales. El total de las muestras completas se ha dividido en 80 % como conjunto de entrenamiento para MTS-GAN y el 20 % como conjunto incompleto que se servirá para evaluar la imputación de datos con este y otros métodos. Los datos faltantes serán agregados al conjunto incompleto, siguiendo dos casos distintos:

- Caso a: Se elimina 20 % de los datos de cada muestra de forma aleatoria.
- Caso b: Se elimina un intervalo de 80 % en cada variable de la muestra.

- Caso c: Se eliminan los datos de la misma forma en que se presentan los datos faltantes en las muestras incompletas de los últimos 5 años de las series de tiempo originales.

En el **caso a** los datos faltantes se presentan de una forma distribuida a lo largo de la muestra, seleccionando un porcentaje de esta para que sean datos faltantes, esta situación intenta replicar la aleatoriedad con la que se presentan los datos faltantes, es una buena opción para probar los métodos de imputación. El **caso b** representa situaciones en las que se tiene un porcentaje de datos faltantes consecutivos. En el **caso c** los datos faltantes siguen la distribución que tiene el conjunto de datos original y por lo tanto, evaluar el desempeño de los métodos de imputación en esta situación ofrecerá una mejor aproximación a los resultados que se obtendrán cuando se realice la imputación sobre el conjunto de datos real. Es importante resaltar la diferencia que existe entre el **caso a** y el **caso c**, ya que el primero solo vuelve faltantes los datos de cada muestra según el porcentaje indicado, mientras que en el segundo los datos se vuelven faltantes tomando como referencia muestras incompletas del conjunto original.

Para evaluar los resultados obtenidos con los diferentes métodos de imputación se utilizarán las métricas descritas en la Tabla 3.2, en donde, y_i representa los valores obtenidos mientras que x_i los valores reales. Estas métricas se aplicaran sobre el vector de salida completo es decir, el vector que contiene las tres variables involucradas.

Métrica	**Definición**	**Ecuación**
MSE	Error cuadrático medio	$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - x_i)^2$
MAE	Error absoluto medio	$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}\lvert y_i - x_i\rvert$
RMSE	Raíz del error cuadrático medio	$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - x_i)^2}$

Tabla 3.2: Definición de métricas utilizadas para evaluar imputación de datos faltantes.

Una vez que se ha realizado la imputación de datos con la estación principal (Merced), se procederá a imputar datos faltantes en algunas de las estaciones disponibles, las cuales se muestran en la Tabla 3.3.

Clave	Nombre
FAC	FES Acatlán
MER	Merced
PED	Pedregal
SAG	San Agustín
TLA	Tlalnepantla
XAL	Xalostoc

Tabla 3.3: Estaciones disponibles para imputación de datos.

La imputación de datos con el modelo MTS-GAN en las diferentes estaciones, se realizará entrenando el modelo con:

- Entrenamiento con datos de la estación Merced: Se imputarán los datos faltantes de todas las estaciones que se encuentren disponibles, entrenando el modelo MTS-GAN con el conjunto completo de la estación Merced, ya que este posee la mayor cantidad de muestras completas.
- Entrenamiento con datos de la propia estación: Se imputarán los datos faltantes de todas las estaciones que se encuentren disponibles, entrenando el modelo MTS-GAN con el conjunto de datos completo de la estación en cuestión.

Al evaluar estos resultados se podrá verificar si el modelo logra generalizar su entrenamiento para otras estaciones diferentes a la que se utilizó para su entrenamiento, o la distribución de éstas es muy distinta, y por lo tanto, es necesario entrenar el modelo con cada una de las estaciones para obtener un buen nivel de precisión.

Una vez que se determine cual será la información de entrenamiento que se utilizará en cada estación, se compararán los resultados obtenido de los modelos MTS-GAN con el de algunos de los métodos de imputación más comunes. Los métodos de imputación que se utilizarán son:

- Imputación de media.
- Imputación de mediana.
- Imputación con K-vecinos cercanos.
- Imputación con MTS-GAN.

3.3.1.2. Imputación sobre conjunto de datos real

En este apartado se realizará la imputación de datos faltantes con los diferentes métodos de imputación sobre las series de tiempo multivariadas originales conformadas por las mediciones comprendidas entre los años 2014 a 2019 (6 años), los cuales son la cantidad de datos que se utilizarán para realizar el pronóstico de las series de tiempo.

En el caso de modelo MTS-GAN se crearán dos conjuntos distintos para realizar la imputación de las series de tiempo multivariadas: **conjunto de datos incompleto** y el **conjuto de datos de entrenamiento**.

El **conjunto de datos incompleto** estará conformado por todas las muestras incompletas que se encuentran entre los años 2014 a 2019, ya que estas son las series que necesitaban la imputación los datos faltantes.

El **conjunto de datos de entrenamiento** para el modelo MTS-GAN estará conformados por todas las muestras completas comprendidas entre los años 1986 a 2020.

Una vez que se han imputados los datos faltantes, se guardaran las series de tiempo reconstruidas con los diferentes métodos de imputación utilizados.

3.3.2. Pronóstico de series de tiempo multivariadas reconstruidas

Una vez que se tengan las series de tiempo multivariadas reconstruidas con los diferentes métodos de imputación propuestos (incluido MTS-GAN), se procederá a realizar el pronóstico de dichas series, haciendo uso de modelos de redes neuronales recurrentes. Los modelos de RNN para realizar el pronóstico son:

- LSTM
- GRU
- BiLSTM
- BiGRU

Todos los modelos se implementaron con la misma configuración, para estar en igualdad de condiciones. Los hiper-parámetros se establecieron probando algunos valores, seleccionando los que ofrecían mejores resultados. Como se puede observar en la Tabla 3.4, donde se muestra los detalles de los modelos utilizados en pronóstico, todos los modelos propuestos contienen la misma configuración, con excepción del tipo de capa en la primera capa, la cual cambia según el modelo en cuestión.

Capa	Tipo	Unidades	Fun. Act.
Capa 1	LSTM, GRU, BiLSTM, BiGRU	32	Sigmoide
Capa 2	MLP	252	Sigmoide

Tabla 3.4: Configuración de los modelos de RNN utilizados para la predicción de las series de tiempo multivariadas.

Para la evaluación a los modelos anteriores se emplearán las mismas métricas utilizadas imputación de datos sobre el conjunto de evaluación, descritas en la Tabla 3.2. Con esto, se podrá determinar qué modelo ofrecen mejores resultados. Además ayudará a averiguar si los pronósticos realizados con las series de tiempo reconstruidas con el modelo MTS-GAN, son más precisos que los pronósticos hechos con las series reconstruidas con algunos de los otros métodos de imputación utilizados.

Las series de tiempo reconstruidas estarán conformadas desde el 1 de Enero del 2014 al 1 de Enero del 2020 y serán divididas en tres conjuntos:

- Conjunto de entrenamiento: Conjunto conformado por el 70 % de la series de tiempo reconstruidas. Del 1 de Enero del 2014 a la 01:00 hasta el 14 de Marzo de 2018 a las 00:00.
- Conjunto de prueba: Conjunto conformado por el 20 % de las series de tiempo reconstruidas. Del 14 de Marzo del 2018 a la 01:00 hasta el 26 de Mayo del 2019 a las 00:00.
- Conjunto de validación: Conjunto conformado por el 10 % de las series de tiempo reconstruidas. Del 26 de Mayo del 2019 a la 01:00 hasta el 1 de Enero del 2020.

Una vez que se ha encontrado el modelo que ofrece los mejores resultados, se seleccionará dicho modelo para entrenar de nuevo y hacer predicciones en todas las series de tiempo sobre el año 2019. La división de las series de tiempo, quedará de la siguiente manera:

- Conjunto de entrenamiento: Conformado por las muestras comprendidas entre el 1 de Enero del 2014 a las 01:00 y el 1 de Enero del 2019 a las 00:00.
- Conjunto de prueba: Conformado por las muestras comprendidas entre 1 de Enero del 2019 a las 01:00 y el 1 de Enero del 2020 a las 00:00

La Figura 3.14 muestra el diagrama de flujo sobre el procedimiento mencionado anteriormente.

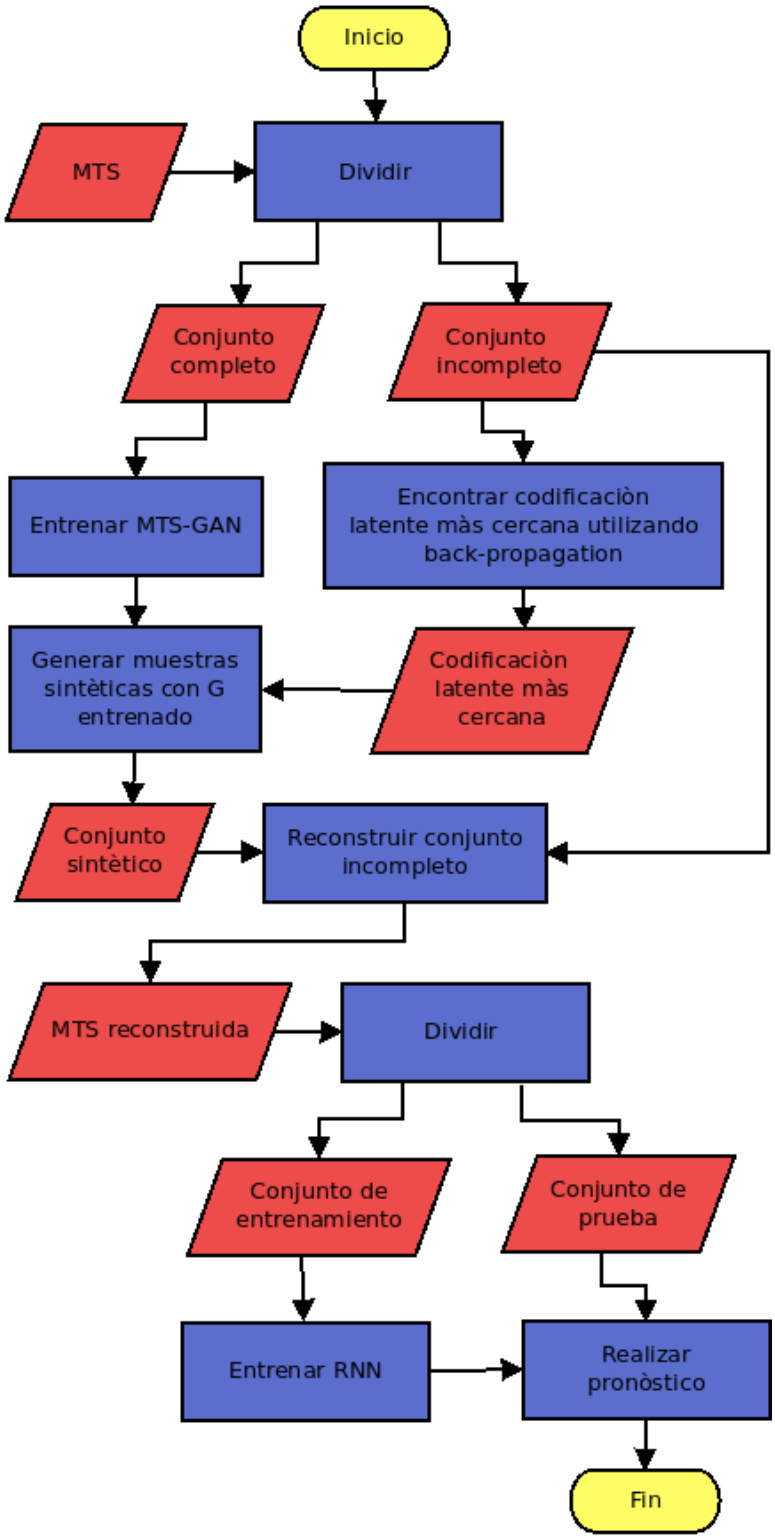

Figura 3.14: Diagrama de flujo.

Capítulo 4

Análisis de resultados

En este capítulo se presentan los resultados obtenidos en la imputación de datos faltantes y el pronóstico sobre las series de tiempo multivariadas de contaminantes del aire en algunas estaciones de la Cuidad de México; en particular la estación Merced. Además se describen los parámetros, configuraciones y pasos de entrenamiento de los modelos propuestos. Al final todos los resultados presentados son contrastados e interpretados para justificar sus valores.

4.1. Resultados de la imputación

A continuación se resumen las distintas etapas que se siguieron para la imputación de datos en el conjunto de evaluación y en el conjunto de datos real. También se incluyen los detalles del pronóstico sobre este último.

4.1.1. Entrenamiento MTS-GAN

Como ya se ha visto anteriormente, el entrenamiento del MTS-GAN, consiste en el entrenamiento de la red generadora del modelo. Para esto se toman todas las muestras disponibles que no presenten ningún dato faltante, ya que de estas muestras el modelo aprenderá la distribución de la serie.

Con el propósito de obtener el mayor número de muestras para el entrenamiento, se utilizaron los datos de la estación Merced, la cual presenta registros desde el año 1986. Las muestras fueron tomadas desde el 1 de Enero de 1986 hasta el 1 de Enero de 2020, obteniendo un total de 5135 muestras completas.

Además de entrenar el modelo con los datos completos de la estación Merced, se hicieron versiones distintas del modelo, entrenándolo con los datos de cada una de las estaciones propuestas. Las estaciones utilizadas son las siguiente:

- FES Acatlán: Muestras tomadas desde 1 de Enero de 1990 hasta 1 de Enero de 2014 (4483 muestras completas).
- Pedregal: Muestras tomadas desde 1 de Enero de 1986 hasta 1 de Enero de 2014 (4819 muestras completas).
- San Agustín: Muestras tomadas desde 1 de Enero de 1994 hasta 1 de Enero de 2014 (3661 muestras completas).
- Tlalnepantla: Muestras tomadas desde 1 de Enero de 1989 hasta 1 de Enero de 2014 (4352 muestras completas).
- Xalostoc: Muestras tomadas desde 1 de Enero de 1987 hasta 1 de Enero de 2014 (4822 muestras completas).

En todos los casos el entrenamiento se realizó durante 250 épocas, utilizando las misma configuración mostrada en [2].

Una vez que se entrenó el modelo, éste era capaz de generar muestras que tuvieran la distribución de la serie de tiempo original a partir de un vector aleatorio. El modelo había aprendido la distribución de la serie de tiempo pero aun era incapaz de hacer la imputación de datos, debido a que los datos que se quieren imputar no pueden ser aleatorios.

4.1.2. Búsqueda de codificación en espacio latente más cercana

En esta etapa se encontró la codificación en el espacio latente más cercana de cada uno de los casos de datos faltantes en los conjuntos de evaluación de las estaciones anteriores. También se encontró la codificación más cercana de los conjuntos reales. Las representaciones obtenidas fueron guardadas para realizar la imputación de datos.

En todos los casos se utilizó back-propagation para encontrar la codificación más cercana. Para optimizar la función de pérdida propuesta en [2] se utilizó la salida del generador entrenado anteriormente con las muestras completas. El proceso se ejecutó durante 1500 pasos de optimización, cuando la función de pérdida tenía un valor significativamente bajo.

Una vez que se obtuvieron las codificaciones, estas fueron guardadas para utilizarlas en la imputación de datos.

4.1.3. Imputación

Por último solo era necesario ingresar las codificaciones más cercanas al generador ya entrenado del modelo para crear una muestra sintética, de donde se obtendrían los

valores faltantes a imputar en la muestra incompleta.

A continuación se muestran los resultados obtenidos en la imputación de datos faltantes en los conjuntos de evaluación de cada estación. Para todos los casos se crearon un total de 1283 muestras incompletas, variando únicamente la cantidad y distribución de los datos faltantes.

4.1.3.1. Resultados de imputación en estación Merced

En las Figuras 4.1, 4.2 y 4.3 se grafican las muestras originales, las muestas incompletas y las muestras reconstruidas de la estación Merced en cada caso propuesto (Caso A, Caso B, Caso C). Donde el eje y indica el valor de la variable y el eje x el tiempo. La **variable 1** (azul) corresponde a los niveles de concentración de ozono, **variable 2** (verde) a la temperatura y la **variable 3** (rojo) a la humedad relativa. Las muestras reconstruidas se obtuvieron con el modelo MTS-GAN entrenado con el conjunto completo de la estación Merced.

En la Tabla 4.1 se encuentran las métricas aplicadas sobre el conjunto original y los conjuntos reconstruidos de la estación Merced. La evaluación de los resultados de la imputación se realizó sobre tres métodos distintos y el modelo MTS-GAN.

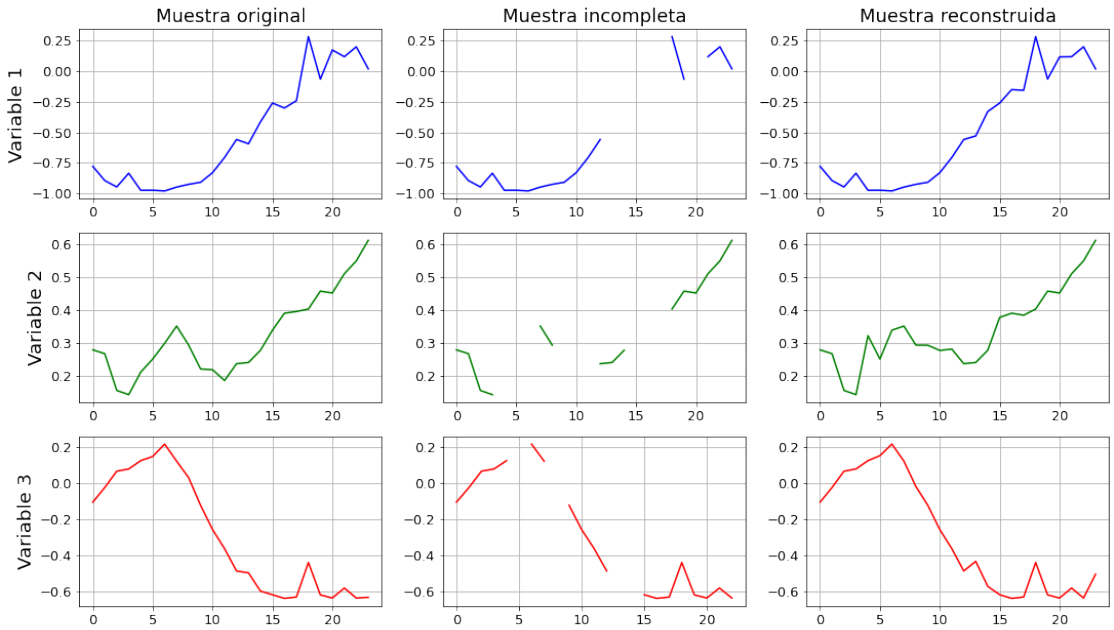

Figura 4.1: Muestra original, incompleta (Caso A) y reconstruida con MTS-GAN de la estación Merced.

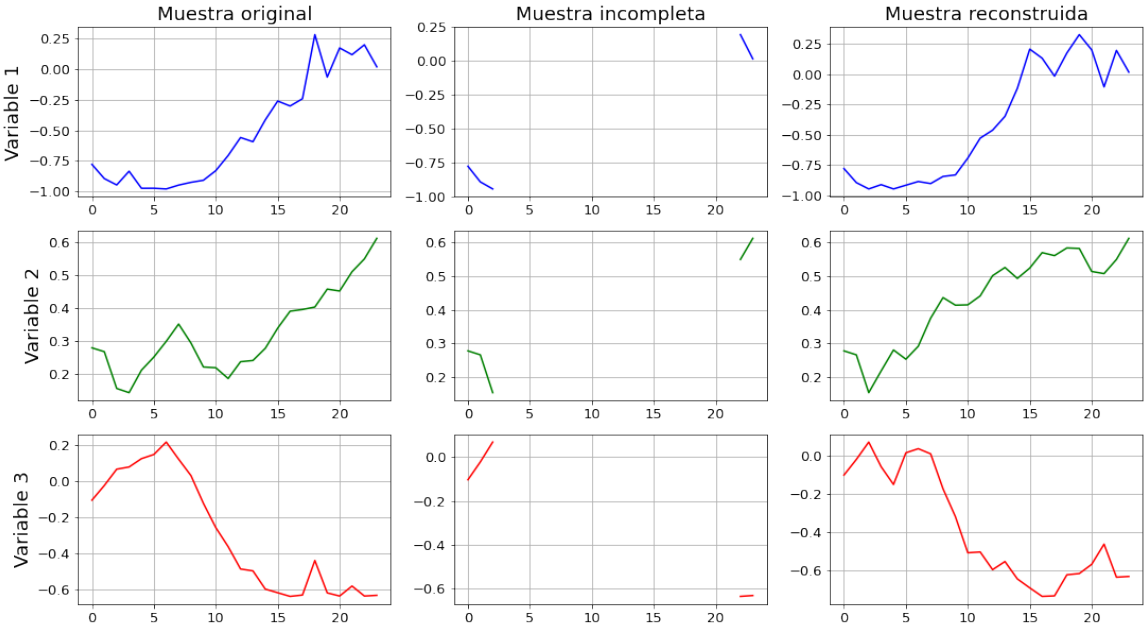

Figura 4.2: Muestra original, incompleta (Caso B) y reconstruida con MTS-GAN de la estación Merced.

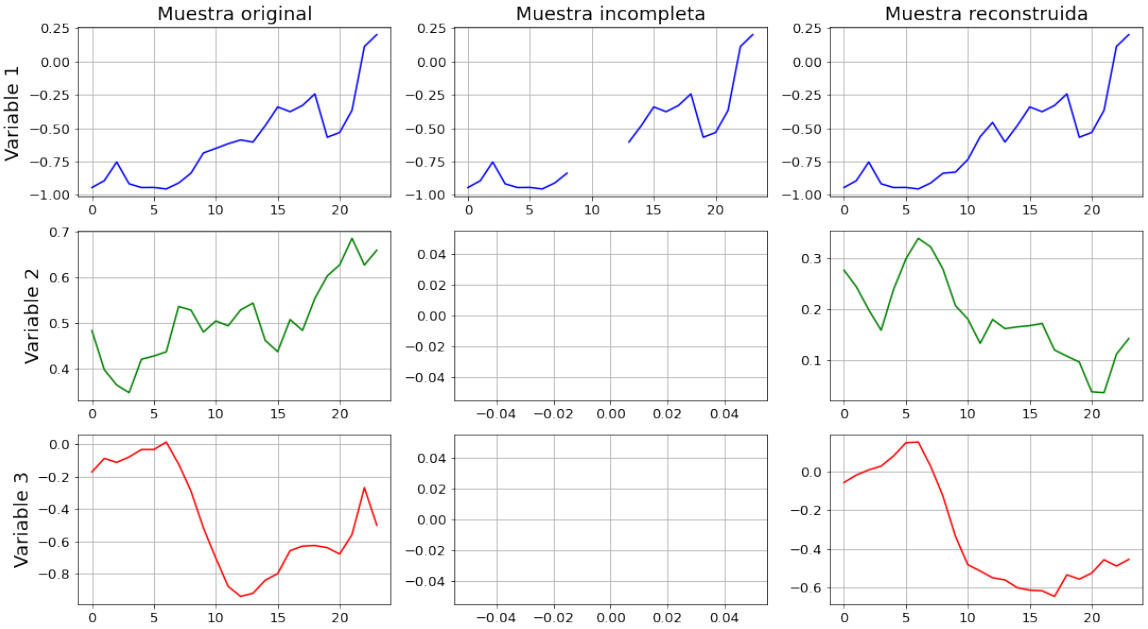

Figura 4.3: Muestra original, incompleta (Caso C) y reconstruida con MTS-GAN de la estación Merced.

Caso	A			B			C		
Método	MAE	MSE	RMSE	MAE	MSE	RMSE	MAE	MSE	RMSE
MTS-GAN	**0.0223**	**0.0045**	**0.0668**	**0.1557**	**0.0583**	**0.2415**	**0.0353**	**0.0144**	**0.1200**
KNN	0.0475	0.0189	0.1374	0.2098	0.0924	0.3039	0.0370	0.0149	0.1221
Valor mediana	0.0522	0.0215	0.1465	0.2157	0.1024	0.3200	0.0392	0.0160	0.1264
Valor medio	0.0524	0.0214	0.1462	0.2098	0.0924	0.3039	0.0399	0.0159	0.1262

Tabla 4.1: Error de reconstrucción en diferentes casos sobre la estación Merced. El **Caso A** contiene datos faltantes aleatorios (20 % en cada muestra), el **Caso B** tiene un intervalo de datos faltantes (80 % en cada muestra) y el **Caso C** presenta datos faltantes como el conjunto de datos original. Los valores en negrita son los mejores resultados.

En los resultados de la imputación en la estación Merced, se muestra que el modelo MTS-GAN logra el mayor nivel de precisión en la reconstrucción del conjunto incompleto para los tres casos propuestos. La Tabla 4.1 muestra para el **Caso A**, MTS-GAN obtiene valores más bajos que el resto de métodos propuestos, la diferencia es notable, incluso con respecto al segundo mejor desempeño dado por KNN. En el **Caso B**, MTS-GAN logra los mejores resultados a pesar de que el número de datos faltantes es mayor. Para el **Caso C**, los resultados siguen siendo mejores con MTS-GAN, aunque la diferencia con respecto al resto de métodos es menor.

4.1.3.2. Resultados de imputación en estación Pedregal

En las Figuras 4.4 y 4.5 se muestran las reconstrucciones de una muestra de la estación Pedregal en cada caso propuesto. Donde el eje y indica el valor de la variable y el eje x el tiempo. La **variable 1** (azul) corresponde a los niveles de concentración de ozono, **variable 2** (verde) a la temperatura y la **variable 3** (rojo) a la humedad relativa. Las muestras reconstruidas se obtuvieron con el modelo MTS-GAN 1 entrenado con el conjunto completo de la estación Pedregal.

En la Tabla 4.2 se encuentran las métricas aplicadas sobre el conjunto original y los conjuntos reconstruidos de la estación Pedregal con cada uno de los métodos propuestos. En este caso se evalúan dos modelos MTS-GAN entrenados con distintos conjuntos de datos; MTS-GAN 1 se ha entrenado con los datos de la estación Pedregal, mientras que MTS-GAN 2 mantiene el entrenamiento con la estación Merced.

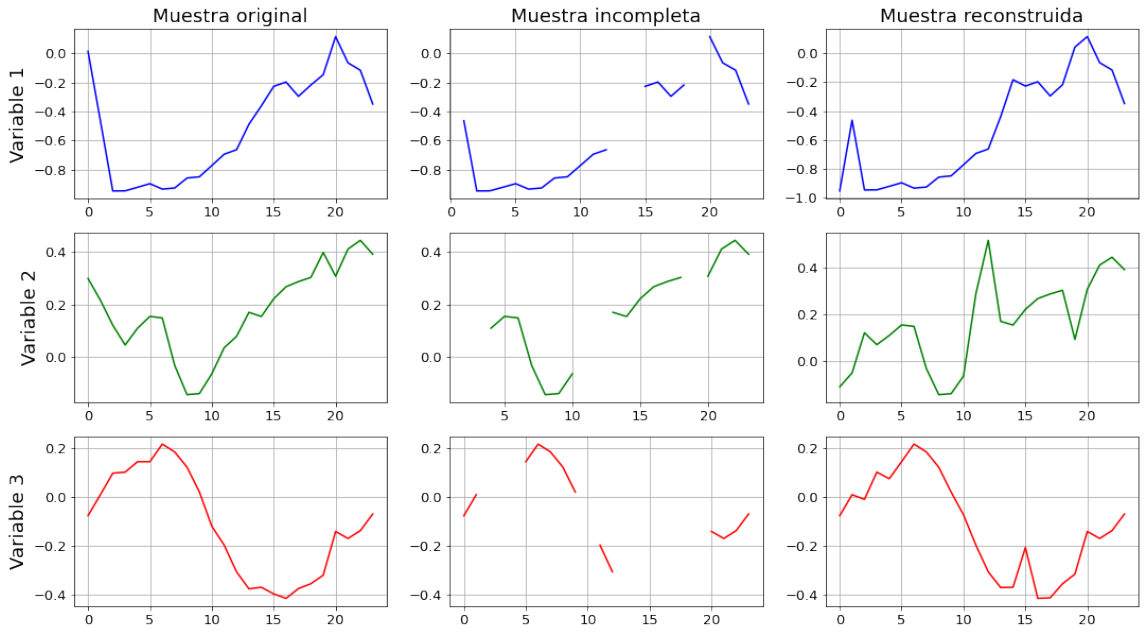

Figura 4.4: Muestra original, incompleta (Caso A) y reconstruida con MTS-GAN 1 de la estación Pedregal.

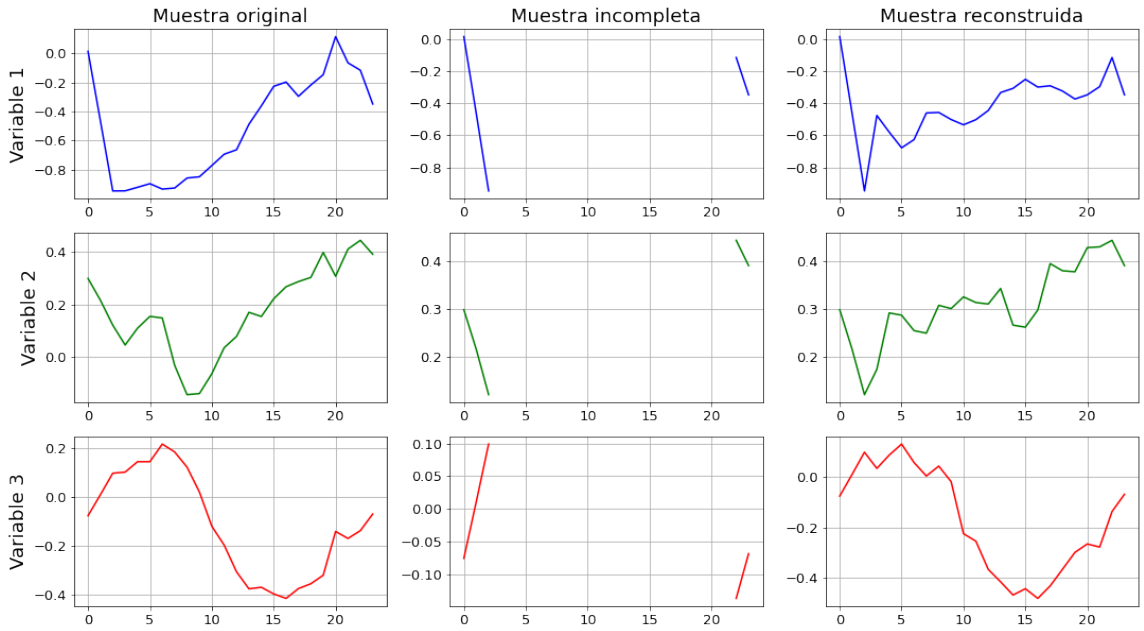

Figura 4.5: Muestra original, incompleta (Caso B) y reconstruida con MTS-GAN 1 de la estación Pedregal.

Caso	A			B			C		
Método	MAE	MSE	RMSE	MAE	MSE	RMSE	MAE	MSE	RMSE
MTS-GAN 1	**0.0244**	**0.0051**	**0.0715**	0.1773	0.0762	0.2760	0.0412	0.0206	0.1434
MTS-GAN 2	0.0259	0.0060	0.0776	**0.1612**	**0.0609**	**0.2468**	0.0379	0.0181	0.1345
KNN	0.0491	0.0203	0.1425	0.2280	0.1035	0.3217	0.0370	0.0164	0.1279
Valor mediana	0.0548	0.0234	0.1530	0.2301	0.1092	0.3304	**0.0347**	0.0142	0.1191
Valor medio	0.0552	0.0232	0.1524	0.2280	0.1035	0.3217	0.0349	**0.0139**	**0.1180**

Tabla 4.2: Error de reconstrucción en diferentes casos sobre la estación Pedregal. El **Caso A** contiene datos faltantes aleatorios (20 % en cada muestra), el **Caso B** tiene un intervalo de datos faltantes (80 % en cada muestra) y el **Caso C** presenta datos faltantes como el conjunto de datos original. Los valores en negrita son los mejores resultados.

Para esta estación la imputación con el modelo MTS-GAN fue la mejor opción en los primeros dos casos. En el **Caso A** los mejores resultados resultados se obtuvieron con MTS-GAN 1, modelo que fue entrenado con el conjunto de datos completo de la estación Pedregal, seguido por MTS-GAN 2, entrenado con el conjunto completo de la estación Merced. Para el **Caso B** los mejores resultados se obtuvieron con MTS-GAN 2, aun así MTS-GAN 1 obtiene los segundos mejores resultados. En ambos casos MTS-GAN 1 y MTS-GAN 2 son mejores opciones para realizar la impitación de datos. Por ultimo, en el **Caso C** MTS-GAN 1 y MTS-GAN 2 no lograron los mejores resultados, sin embargo, la diferencia con respecto al resto de métodos no es tan pronunciada.

4.1.3.3. Resultados de imputación en estación Xalostoc

En las Figuras 4.6 y 4.7 se muestran las reconstrucciones de una muestra de la estación Xalostoc en cada caso propuesto. Donde el eje y indica el valor de la variable y el eje x el tiempo. La **variable 1** (azul) corresponde a los niveles de concentración de ozono, **variable 2** (verde) a la temperatura y la **variable 3** (rojo) a la humedad relativa. Las muestras reconstruidas se obtuvieron con el modelo MTS-GAN 1 entrenado con el conjunto completo de la estación Xalosctoc.

En la Tabla 4.3 se encuentran las métricas aplicadas sobre el conjunto original y los conjuntos reconstruidos de la estación Xalostoc con cada uno de los métodos propuestos. En este caso se evalúan dos modelos MTS-GAN entrenados con distintos conjuntos de datos. MTS-GAN 1 se ha entrenado con los datos de la estación Xalostoc, mientras que MTS-GAN 2 mantiene el entrenamiento con la estación Merced.

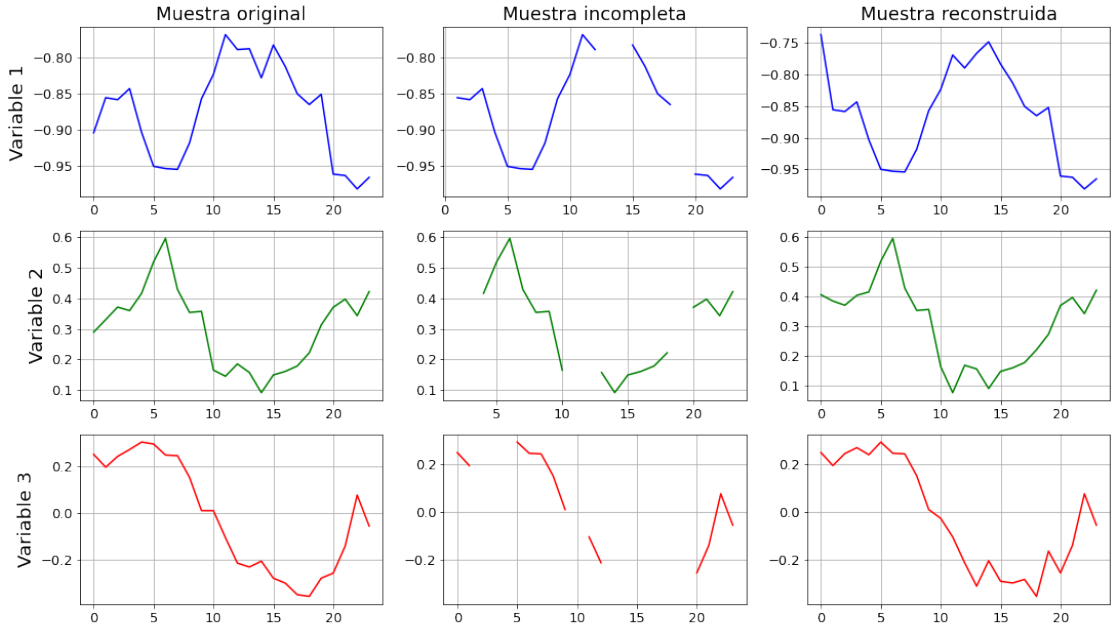

Figura 4.6: Muestra original, incompleta (Caso A) y reconstruida con MTS-GAN 1 de la estación Xalostoc.

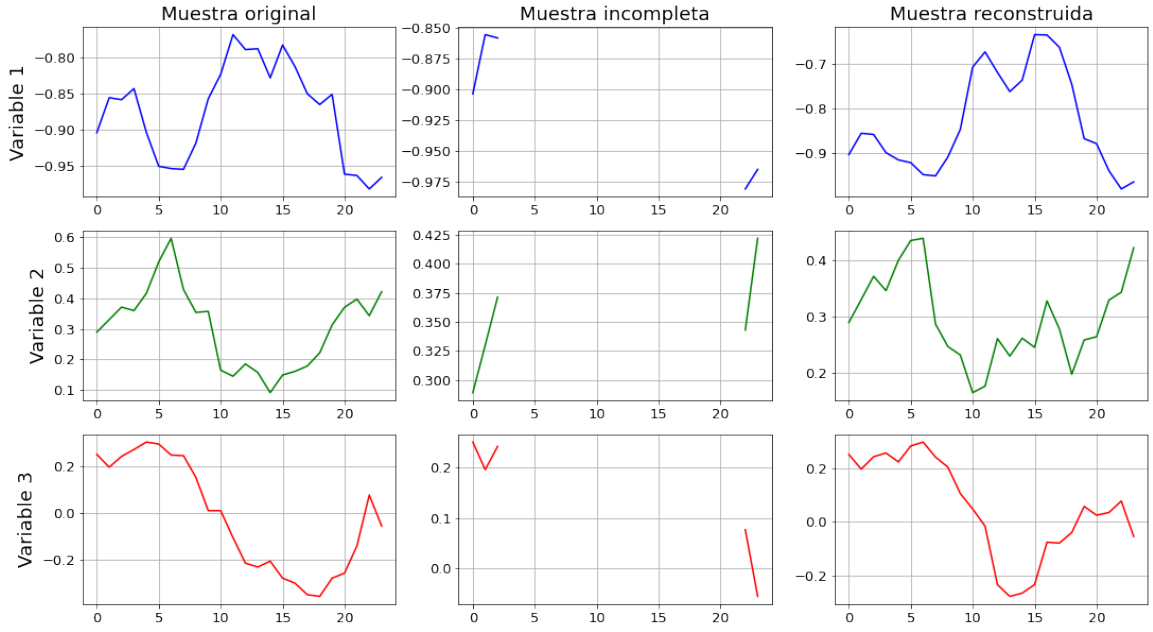

Figura 4.7: Muestra original, incompleta (Caso B) y reconstruida con MTS-GAN 1 de la estación Xalostoc.

Caso	A			B			C		
Método	MAE	MSE	RMSE	MAE	MSE	RMSE	MAE	MSE	RMSE
MTS-GAN 1	**0.0178**	**0.0027**	**0.0518**	0.1441	0.0544	0.2332	**0.0349**	**0.0122**	**0.1103**
MTS-GAN 2	0.0202	0.0036	0.0604	**0.1440**	**0.0509**	**0.2255**	0.0371	0.0140	0.1185
KNN	0.0436	0.0165	0.1284	0.1953	0.0784	0.2800	0.0426	0.0166	0.1288
Valor mediana	0.0481	0.0185	0.1359	0.1976	0.0851	0.2917	0.0434	0.0164	0.1282
Valor medio	0.0484	0.0183	0.1352	0.1953	0.0784	0.2800	0.0441	0.0165	0.1283

Tabla 4.3: Error de reconstrucción en diferentes casos sobre la estación Xalostoc. El **Caso A** contiene datos faltantes aleatorios (20 % en cada muestra), el **Caso B** tiene un intervalo de datos faltantes (80 % en cada muestra) y el **Caso C** presenta datos faltantes como el conjunto de datos original. Los valores en negrita son los mejores resultados.

En la estación Xalostoc los mejores resultados de la imputación se obtienen con los modelos MTS-GAN. En el **Caso A** y **Caso C** los mejores resultados se obtuvieron con el modelo MTS-GAN 1, entrenado con el conjunto completo de la estación Xalostoc, seguido por MTS-GAN 2 entrenado con el conjunto completo de la estación Merced. Para el **Caso B** el mejor resultado se obtuvo con MTS-GAN 2. La diferencia entre las métricas de los modelos MTS-GAN 1 y MTS-GAN 2 es muy reducida, por lo tanto, ambos modelos son viables para imputar datos faltantes en la estación Xalostoc.

4.1.3.4. Resultados de imputación en estación Tlalnepantla

En las Figuras 4.8 y 4.9 se muestran las reconstrucciones de una muestra de la estación Tlalnepantla en cada caso propuesto. Donde el eje y indica el valor de la variable y el eje x el tiempo. La **variable 1** (azul) corresponde a los niveles de concentración de ozono, **variable 2** (verde) a la temperatura y la **variable 3** (rojo) a la humedad relativa. Las muestras reconstruidas se obtuvieron con el modelo MTS-GAN 1 entrenado con el conjunto completo de la estación Tlalnepantla.

En la Tabla 4.4 se encuentran las métricas aplicadas sobre el conjunto original y los conjuntos reconstruidos de la estación Tlalnepantla con cada uno de los métodos propuestos. En este caso se evalúan dos modelos MTS-GAN entrenados con distintos conjuntos de datos. MTS-GAN 1 se ha entrenado con los datos de la estación Tlalnepantla, mientras que MTS-GAN 2 mantiene el entrenamiento con la estación Merced.

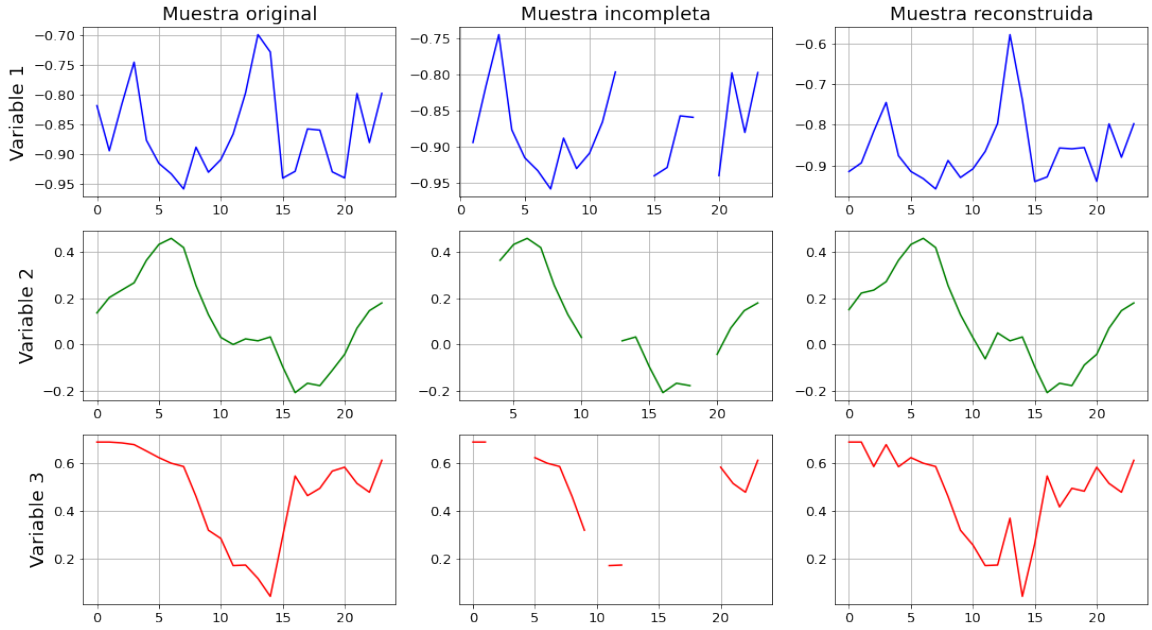

Figura 4.8: Muestra original, incompleta (Caso A) y reconstruida con MTS-GAN 1 de la estación Tlalnepantla.

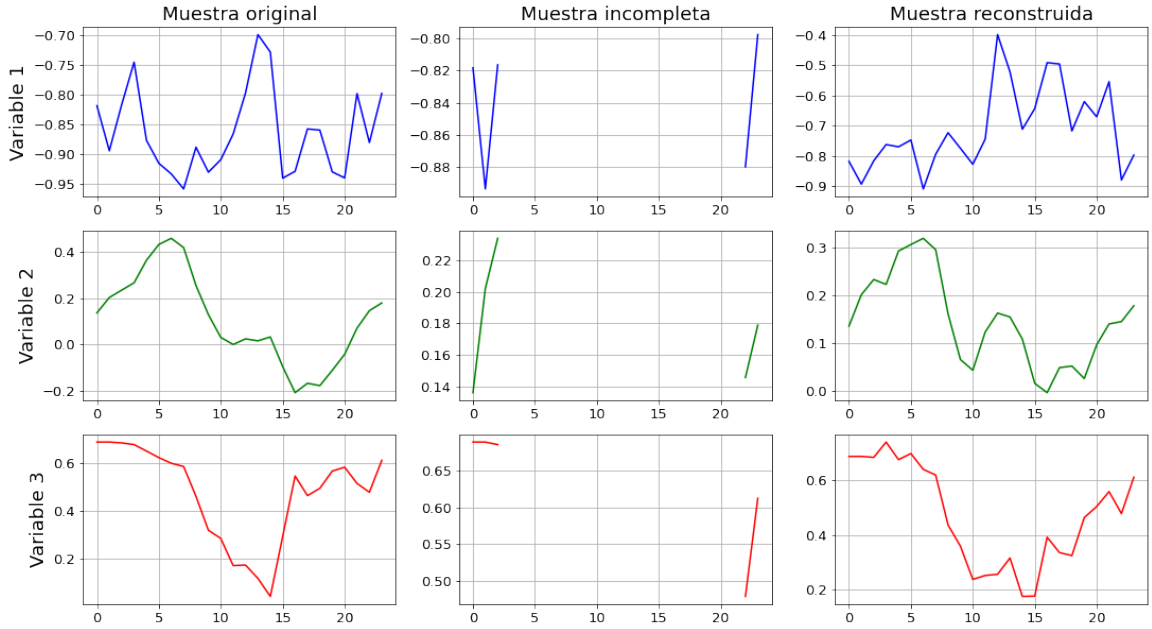

Figura 4.9: Muestra original, incompleta (Caso B) y reconstruida con MTS-GAN 1 de la estación Tlalnepantla.

Caso	A			B			C		
Método	MAE	MSE	RMSE	MAE	MSE	RMSE	MAE	MSE	RMSE
MTS-GAN 1	**0.0180**	**0.0030**	**0.0546**	0.1564	0.0682	0.2611	0.0525	0.0239	0.1547
MTS-GAN 2	0.0213	0.0042	0.0644	**0.1480**	**0.0547**	**0.2338**	**0.0435**	**0.0169**	**0.1300**
KNN	0.0464	0.0188	0.1370	0.2113	0.0936	0.3059	0.0468	0.0181	0.1346
Valor mediana	0.0515	0.0212	0.1456	0.2160	0.1038	0.3221	0.0475	0.0178	0.1333
Valor medio	0.0518	0.0210	0.1449	0.2113	0.0936	0.3059	0.0478	0.0175	0.1324

Tabla 4.4: Error de reconstrucción en diferentes casos sobre la estación Tlalnepantla. El **Caso A** contiene datos faltantes aleatorios (20 % en cada muestra), el **Caso B** tiene un intervalo de datos faltantes (80 % en cada muestra) y el **Caso C** presenta datos faltantes como el conjunto de datos original. Los valores en negrita son los mejores resultados.

Los mejores resultados se obtuvieron con los modelos MTS-GAN. En **Caso A** Los mejores resultados siguieron siendo obtenidos por el modelos MTS-GAN, entrenado con el conjunto completo de la estación Tlalnepantla. Para **Caso B** y **Caso C** los mejores resultados se obtuvieron con el modelos MTS-GAN 2, entrenado con el conjunto completo de la estación Merced.

4.1.3.5. Resultados de imputación en estación FES Acatlán

En las Figuras 4.10 y 4.11 se muestran las reconstrucciones de una muestra de la estación FES Acatlán en cada caso propuesto. Donde el eje y indica el valor de la variable y el eje x el tiempo. La **variable 1** (azul) corresponde a los niveles de concentración de ozono, **variable 2** (verde) a la temperatura y la **variable 3** (rojo) a la humedad relativa. Las muestras reconstruidas se obtuvieron con el modelo MTS-GAN 1 entrenado con el conjunto completo de la estación FES Acatlán.

En la Tabla 4.5 se encuentran las métricas aplicadas sobre el conjunto original y los conjuntos reconstruidos de la estación FES Acatlán con cada uno de los métodos propuestos. En este caso se evalúan dos modelos MTS-GAN entrenados con distintos conjuntos de datos. MTS-GAN 1 se ha entrenado con los datos de la estación FES Acatlán, mientras que MTS-GAN 2 mantiene el entrenamiento con la estación Merced.

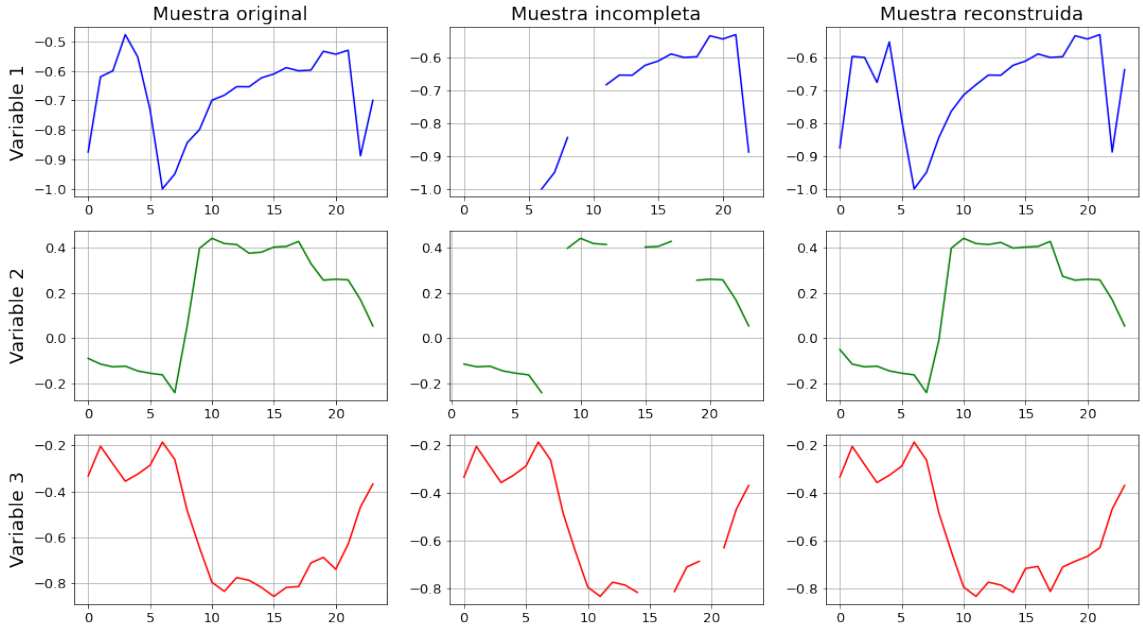

Figura 4.10: Muestra original, incompleta (Caso A) y reconstruida con MTS-GAN 1 de la estación FES Acatlán.

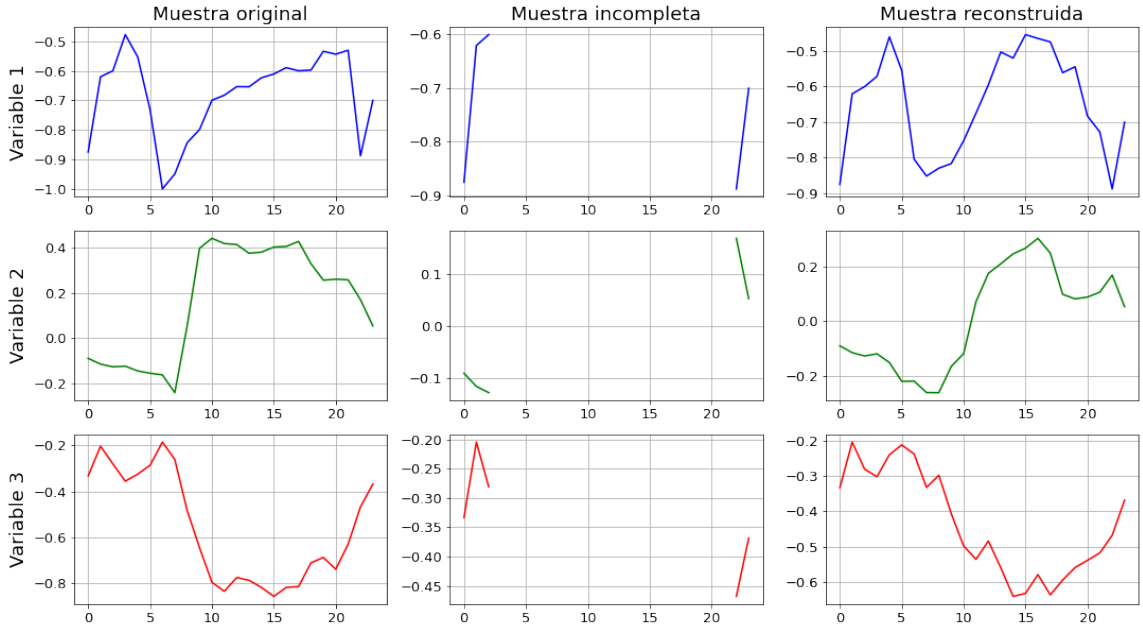

Figura 4.11: Muestra original, incompleta (Caso B) y reconstruida con MTS-GAN 1 de la estación FES Acatlán.

Caso	A			B			C		
Método	MAE	MSE	RMSE	MAE	MSE	RMSE	MAE	MSE	RMSE
MTS-GAN 1	**0.0207**	**0.0036**	**0.0601**	**0.1814**	**0.0851**	**0.2917**	**0.0957**	0.0485	0.2202
MTS-GAN 2	0.0316	0.0082	0.0904	0.2009	0.0888	0.2979	0.0984	0.0493	0.2220
KNN	0.0487	0.0206	0.1435	0.2657	0.1412	0.3758	0.0976	**0.0475**	**0.2179**
Valor mediana	0.0603	0.0284	0.1684	0.2642	0.1469	0.3833	0.1042	0.0521	0.2283
Valor medio	0.0607	0.0280	0.1674	0.2657	0.1412	0.3758	0.1049	0.0515	0.2270

Tabla 4.5: Error de reconstrucción en diferentes casos sobre la estación FES Acatlán. El **Caso A** contiene datos faltantes aleatorios (20 % en cada muestra), el **Caso B** tiene un intervalo de datos faltantes (80 % en cada muestra) y el **Caso C** presenta datos faltantes como el conjunto de datos original. Los valores en negrita son los mejores resultados.

En la estación FES Acatlán, los modelos MTS-GAN (1 y 2) mostraron buen desempeño. En el **Caso A** y **Caso B** los mejores resultados en la imputación se lograron con el modelos MTS-GAN 1. En el **Caso C** la métrica MAE es más baja en el modelo MTS-GAN 1, sin embargo las demás métricas mostraron que el modelo KNN fue mejor, aunque, la diferencia entre estos modelos, no es muy grande.

4.1.3.6. Resultados de imputación en estación San Agustín

En las Figuras 4.12 y 4.13 se muestran las reconstrucciones de una muestra de la estación San Agustín en cada caso propuesto. Donde el eje y indica el valor de la variable y el eje x el tiempo. La **variable 1** (azul) corresponde a los niveles de concentración de ozono, **variable 2** (verde) a la temperatura y la **variable 3** (rojo) a la humedad relativa. Las muestras reconstruidas se obtuvieron con el modelo MTS-GAN 1 entrenado con el conjunto completo de la estación San Agustín.

En la Tabla 4.6 se encuentran las métricas aplicadas sobre el conjunto original y los conjuntos reconstruidos de la estación San Agustín, en donde se utilizaron tres métodos distintos y el modelo MTS-GAN. En este caso se evalúan dos modelos MTS-GAN. El modelo MTS-GAN 1 se ha entrenado con los datos de la estación San Agustín, mientras que MTS-GAN 2 mantiene el entrenamiento con la estación Merced.

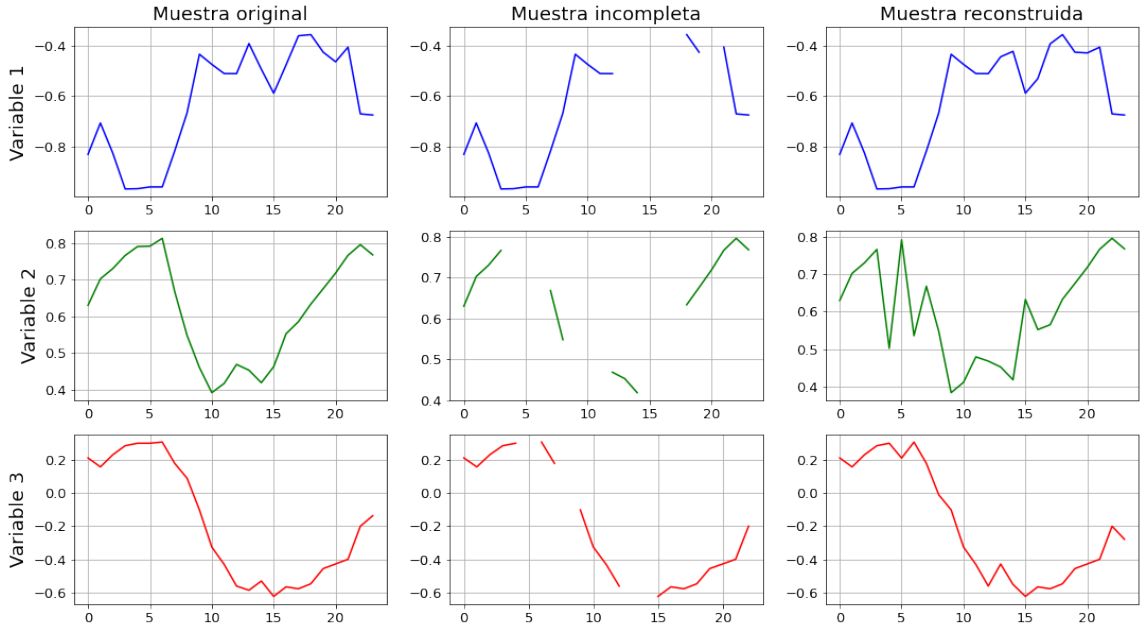

Figura 4.12: Muestra original, incompleta (Caso A) y reconstruida con MTS-GAN 1 de la estación San Agustín.

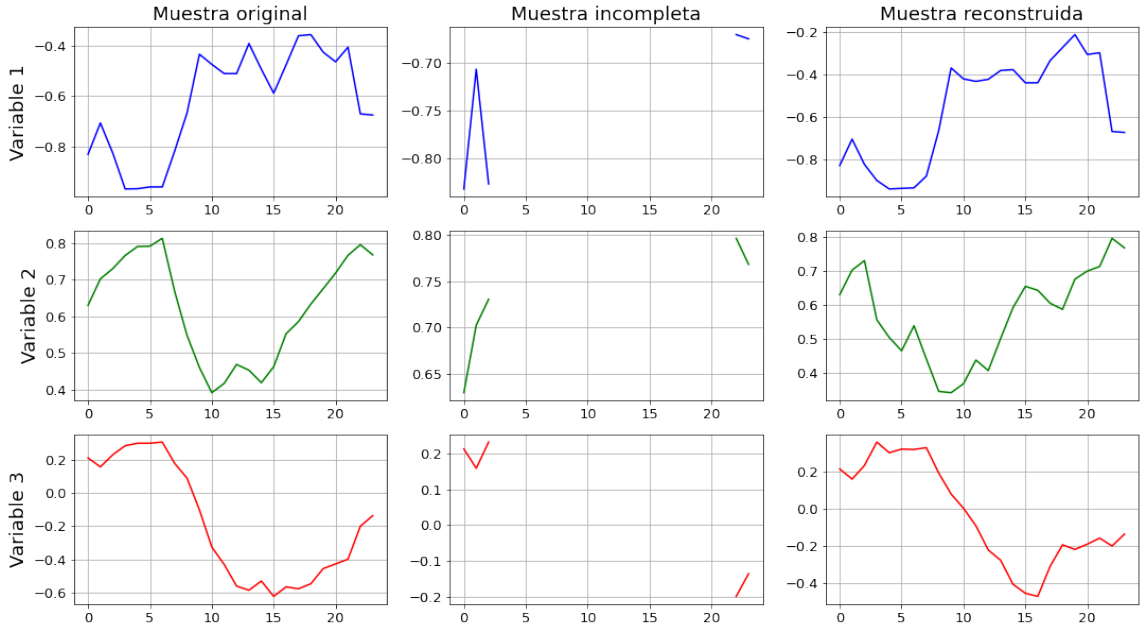

Figura 4.13: Muestra original, incompleta (Caso B) y reconstruida con MTS-GAN 1 de la estación San Agustín.

Caso	A			B			C		
Método	MAE	MSE	RMSE	MAE	MSE	RMSE	MAE	MSE	RMSE
MTS-GAN 1	**0.0228**	**0.0046**	**0.0680**	**0.1422**	**0.0462**	**0.2149**	**0.0781**	**0.0366**	**0.1912**
MTS-GAN 2	0.0240	0.0051	0.0713	0.1550	0.0561	0.2369	0.0800	0.0396	0.1990
KNN	0.0468	0.0191	0.1381	0.2133	0.0963	0.3103	0.0795	0.0390	0.1975
Valor mediana	0.0516	0.0214	0.1464	0.2155	0.1016	0.3187	0.0852	0.0430	0.2074
Valor medio	0.0519	0.0213	0.1458	0.2133	0.0963	0.3103	0.0834	0.0404	0.2011

Tabla 4.6: Error de reconstrucción en diferentes casos sobre la estación San Agustín. El **Caso A** contiene datos faltantes aleatorios (20 % en cada muestra), el **Caso B** tiene un intervalo de datos faltantes (80 % en cada muestra) y el **Caso C** presenta datos faltantes como el conjunto de datos original. Los valores en negrita son los mejores resultados.

En la estación San Agustín, en todos los casos se obtuvo la mejor precisión en la imputación de datos con el modelo MTS-GAN 1, entrenado con el conjunto completo de la estación San Agustín, seguido por el modelo MTS-GAN 2, excepto en el **Caso C**, donde los segundo mejores resultados se obtuvieron con el modelo KNN, aunque la diferencia es muy pequeña. Por lo tanto, ambos modelos MTS-GAN muestran se opciones bastante precisas para realizar la imputación de datos en esta estación.

4.1.4. Análisis de resultados en la imputación de datos

Los resultados de la imputación en las estaciones anteriores son similares. En el **Caso A**, en donde los datos faltantes son aleatorios (aproximadamente 20 % de cada muestra), todas la estaciones obtuvieron los mejores resultados con el modelo MTS-GAN, específicamente MTS-GAN 1, el cual fue entrenado con el conjunto completo de la estación en cuestión. El segundo mejor resultado fue en todas las estaciones, el modelo MTS-GAN 2, que fue entrenado con el conjunto completo de la estación Merced. Por lo tanto el modelo MTS-GAN es mejor realizando la imputación de datos en los casos donde los datos faltante se presentan de forma aleatoria a lo largo de la muestra.

Para el **Caso B**, en donde los datos faltantes se presentan de forma continua (aproximadamente 80 % de cada muestra), MTS-GAN 1 y MTS-GAN 2 obtuvieron los mejores resultados en todas las estaciones, existiendo una pequeña diferencia entre estos dos modelos. En este caso, el modelo MTS-GAN es mejor para realizar la imputación de datos, incluso con muestras en donde los datos faltantes se presentan en un intervalo del 80 % del tamaño de la muestra.

En ambos casos, el modelo MTS-GAN es capaz de capturar la distribución de las series de tiempo y hacer la imputación con un mayor nivel de precisión que el resto de métodos que se presentan.

En el último caso, el **Caso C**, en donde los datos faltantes se presentan, como aparecen en el conjunto de datos original. La diferencia entre la precisión de los modelos MTS-GAN 1 y MTS-GAN 2 con el resto de métodos, es menor, aun así, en la mayoría de las estaciones los mejores resultados son obtenidos por MTS-GAN.

A diferencia de los dos casos anteriores, este caso presenta una situación diferente, y es que existen muestras del conjunto incompleto, que tienen todas las mediciones de alguna variable vacías. Esto representa un problema para el modelo MTS-GAN, ya que al momento de buscar la codificación en el espacio latente más cercana, se dificulta la capacidad del modelo para buscar relaciones entre las variables disponibles, debido a que al menos una está totalmente vacía, sin ningún dato que sirva como referencia para encontrar la codificación más cercana de dicha variable. Los datos sintéticos generados para imputar en dicha variable siguen la distribución del conjunto de la serie de tiempo con que el modelo fue entrenado, pero no es tan preciso que cuando se tiene algo de información. La precisión de imputación es menor cuando aumenta el número de variables vacías.

A continuación se muestran dos ejemplos que ilustran el problema descrito anteriormente. Las Figuras 4.14 y 4.15 muestran la imputación de datos con el modelo MTS-GAN sobre una misma muestra aleatoria de la estación Merced. La Figura 4.14 corresponde a la reconstrucción de una muestra incompleta que contiene dos variables totalmente vacías. Mientras que en la Figura 4.15 se puede observar la reconstrucción de una muestra incompleta que tiene cuatro valores en las variables que estaban vacías anteriormente.

A pesar de que el modelo hace un buen intento por realizar una aproximación a la forma original de las variables faltantes en la serie de tiempo, las magnitudes de la muestra reconstruida son más parecidas a los de la muestra original cuando se tienen algunos valores en las variables que tienen datos faltantes que cuando se encuentran totalmente vacías. Dicha mejoría es visible en las gráficas de las muestras reconstruidas y en las métricas de error.

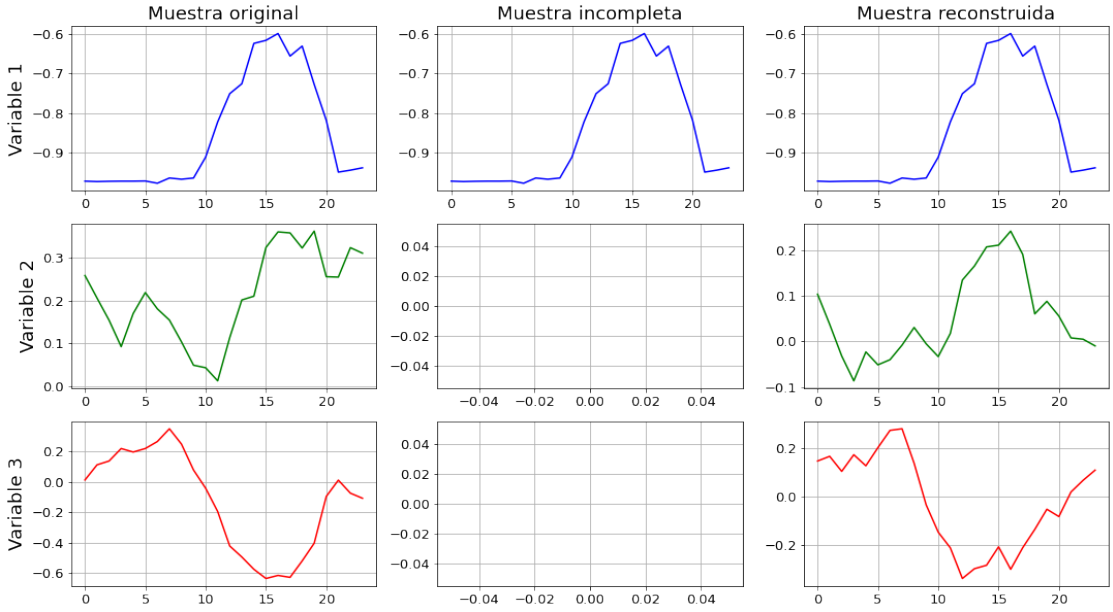

Figura 4.14: Ejemplo de imputación de datos sobre una muestra de la estación Merced con dos variables totalmente vacías. Los valores de las métricas de error son: MAE= 0.1667, MSE=0.0596 y RMSE=0.2441

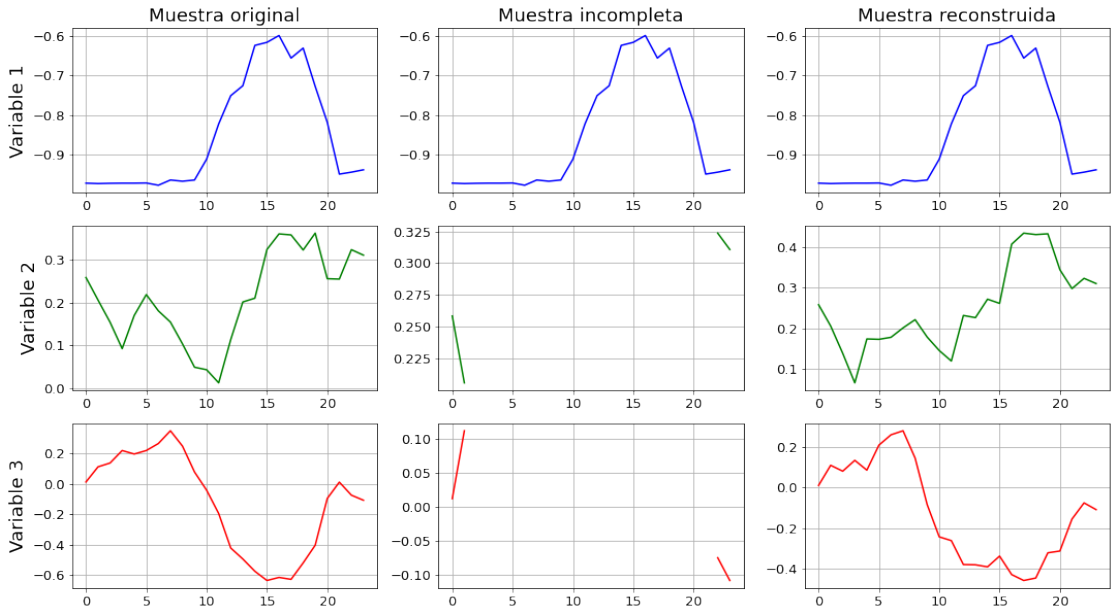

Figura 4.15: Ejemplo de imputación de datos sobre una muestra de la estación Merced con dos variables únicamente con cuatro valores. Los valores de las métricas de error son: MAE=0.0442, MSE=0.0062 y RMSE=0.0790

Esto indica que que este modelo logra un mayor nivel de precisión en la reconstrucción de las muestras.

En general el modelo MTS-GAN hace buen trabajo para hacer la reconstrucción de las muestras incompletas en cualquiera de los tres casos presentados en este trabajo. Sin embargo, aun existen algunos puntos a mejorar, ya que en algunos casos las señales reconstruidas presentan picos y/o valles en los intervalos imputados, los cuales no están presentes en la señal original. Estas diferencias pueden presentarse debido a que la codificación en el espacio latente que se encontró mediante back-propagation no es lo suficientemente precisa, lo cual se traduce como muestras sintéticas no tan buenas y por lo tanto, los datos imputados en las muestras reconstruidas presentan algunas irregularidades con respecto a las muestras originales. Una de las causas por las que se dificulta encontrar una codificación precisa, es que durante la optimización de back-propagation es fácil que este quede atascado en puntos locales, evitando que se encuentre la codificación más cercana [2]. Para reducir este problema, se calculó la codificación en el espacio latente más cercana con diferentes valores iniciales, seleccionando la que mostró un error más pequeño. Aun así es importante mencionar que todavía pueden obtenerse mejores resultados con diferentes valores iniciales que permitan encontrar una codificación de las muestras incompletas más precisa.

4.2. Resultados del pronóstico

En presentacieste apartado se encuentran los resultados de los pronósticos sobre los conjuntos de datos reales de las series de tiempo de contaminantes del aire. Estas series de tiempo fueron reconstruidas utilizando el modelo MTS-GAN, el cual fue entrenado con los datos completos de la estación que se necesitaba reconstruir.

En primer lugar, se realizó el pronóstico sobre las series de tiempo multivariadas de la estación Merced, evaluando su desempeño sobre un conjunto de validación y prueba. Una vez que se determinó cual de estos modelos consiguió los mejores resultados, se utilizó para realizar un pronóstico en el resto de estaciones, utilizando el modelo entrenado con los datos de la estación Merced.

Antes de comenzar con el pronóstico, se tomó cada serie de tiempo reconstruida, la cual incluía las variables: ozono, temperatura y humedad relativa, y se le agregaron características (variables) que serían útiles para el modelo de pronóstico. Dichas variables fueron componente seno de día, componente coseno de día, componente seno de año y componente coseno de año, tal y como se vio en el Capítulo 4. Las variables resultantes, se pueden observar en la Tabla 4.7.

Número	**Variable**	**Clave**
1	Ozono (ppb)	O3
2	Temperatura (°C)	TMP
3	Humedad relativa (%)	RH
4	Componente seno día	DSIN
5	Componente coseno día	DCOS
6	Componente seno año	YSIN
7	Componente coseno año	YCOS

Tabla 4.7: Variables de las series de tiempo multivariadas utilizadas para el pronóstico.

4.2.1. Evaluando modelos RNNs

Los modelos de RNNs propuestos para realizar el pronósticos de las series de tiempo multivariadas son:

- LSTM
- GRU
- BiLSTM
- BiGRU

El conjunto de datos reales de la estación Merced, está conformado por los registros de ozono, temperatura y humedad relativa, desde el 1 de Enero de 2014 hasta 1 Enero de 2020. La serie de tiempo multivariada obtenida fue divida en 3 conjuntos:

- Entrenamiento: Conjunto conformado por el 70 % del conjunto original, destinado a entrenar los modelos propuestos para el pronóstico de las series de tiempo.
- Validación: Conjunto conformado por el 20 % del conjunto original, utiliza parte del conjunto de entrenamiento para seleccionar el mejor modelo, al evaluar el desempeño de varios modelos sobre este conjunto.
- Prueba: Conjunto conformado por el 10 % del conjunto original, utilizado para evaluar el modelo final.

Los resultados sobre el conjunto de validación y de prueba se muestran en la Tabla 4.8 y 4.9 respectivamente..

Modelo	**RMSE**	**MSE**	**MAE**
LSTM	0.3309	0.1095	0.1616
GRU	0.3312	0.1097	0.1617
BiLSTM	0.3267	0.1067	0.1599
BiGRU	**0.3223**	**0.1039**	**0.1519**

Tabla 4.8: Resusltados del pronónostico en la estación Merced sobre el conjunto de validación.

Modelo	RMSE	MSE	MAE
LSTM	0.3044	0.0927	0.1491
GRU	0.3051	0.0931	0.1492
BiLSTM	0.3018	0.0911	0.1483
BiGRU	**0.2980**	**0.0888**	**0.1397**

Tabla 4.9: Resusltados del pronónostico en la estación Merced sobre el conjunto de prueba.

En los resultados anteriores, se puede apreciar que existen modelos que ofrecen mejor rendimiento que otros para esta tarea, hasta llegar al modelo BiGRU, que mostró los mejores resultados. Sin embargo, esto no significa que este modelo sea el mejor para todas las tareas de pronóstico, ya que, dependiendo de la aplicación, un modelo LSTM o GRU puede hacer mejor trabajo. Dados los resultados obtenidos, se optó por utilizar el modelo BiGRU para la realizar el pronóstico de las series de tiempo, tanto de la estación Merced, como del las otras cinco estaciones.

4.2.2. Pronóstico

Una vez que se escogió el modelo BiGRU para realizar el pronóstico de las series de tiempo reconstruidas, se tomó de nuevo el conjunto completo de la estación Merced, es decir, el conjunto conformado desde el 1 de Enero de 2014 hasta el 1 de Enero de 2020 y se formaron nuevos conjuntos.

- Conjunto de entrenamiento: Conformado por todas las muestras de la estación Merced, comprendidas entre el 1 de Enero del 2014 a las 01:00, hasta el 1 de Enero del 2019 a las 00:00. Conjunto utilizado para entrenar el modelo BiGRU.
- Conjunto de prueba: Conjunto utilizado para evaluar modelo BiGRU, conformado por todas las muestra entre el 1 de Enero de 2019 a las 01:00 hasta el 1 de Enero de 2020 a las 12:00. Se realizaron en total, 6 conjuntos de prueba, uno por cada estación de monitoreo.

El modelo BiGRU, se entrenó únicamente con la series de tiempo reconstruidas con el modelo MTS-GAN de la estación Merced y fue utilizado para realizar el pronóstico tanto en el conjunto de prueba de la estación Merced, como en el conjunto de prueba de las demás estaciones seleccionadas.

4.2.2.1. Resultados del pronóstico en la estación Merced

Se utilizó el modelo BiGRU entrenado con las series de tiempo de las estación Merced para realizar el pronóstico sobre el conjunto de prueba de la misma estación.

La Figura 4.16 se muestra la predicción sobre un día de la serie de tiempo multivariada de la estación Merced, donde se da como entrada las mediciones del 1 de enero del 2019 a la 1:00 hasta el 2 de Enero del 2019 a las 0:00 y se obtiene el pronóstico del 1 de Enero de 2019 a las 12:00 hasta el 2 de Enero de 2019 a las 00:00 (36 horas).

La Tabla 4.10 muestra las métricas de error sobre los resultados obtenidos en el pronóstico sobre el conjunto de prueba de la estación Merced.

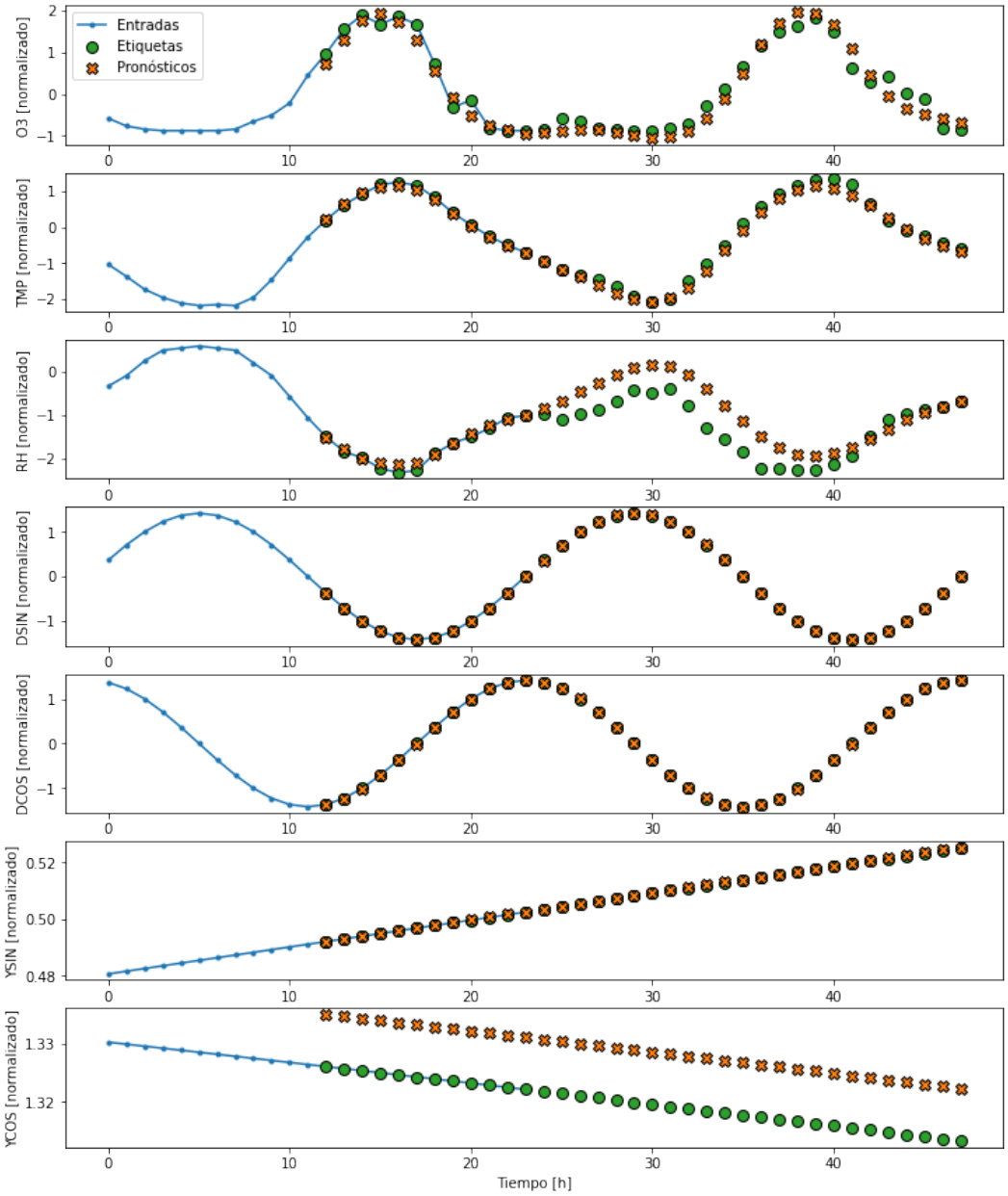

Figura 4.16: Entradas, etiquetas y pronóstico sobre muestra de la serie de tiempo multivariada de la estación Merced.

Método de imputación	MAE	MSE	RMSE
MTS-GAN	**0.1001**	**0.0462**	**0.2157**
KNN	0.1158	0.0545	0.2335
Valor medio	0.1196	0.0612	0.2474

Tabla 4.10: Resultados del pronóstico sobre la estación Merced, entrenando el modelo BiGRU con datos de la estación Merced reconstruidos con MTS-GAN, KNN y valor medio. Los valores en negrita indican los mejores resultados.

4.2.2.2. Resultados del pronóstico en la estación Pedregal

Se utilizó el modelo BiGRU entrenado con las series de tiempo de las estación Merced para realizar el pronóstico sobre el conjunto de prueba de la estación Pedregal.

La Figura 4.17 se muestra la predicción sobre un día de la serie de tiempo multivariada de la estación Pedregal, donde se da como entrada las mediciones del 1 de enero del 2019 a la 1:00 hasta el 2 de Enero del 2019 a las 0:00 y se obtiene el pronóstico del 1 de Enero de 2019 a las 12:00 hasta el 2 de Enero de 2019 a las 00:00 (36 horas).

La Tabla 4.11 muestra las métricas de error sobre los resultados obtenidos en el pronóstico sobre el conjunto de prueba de la estación Pedregal.

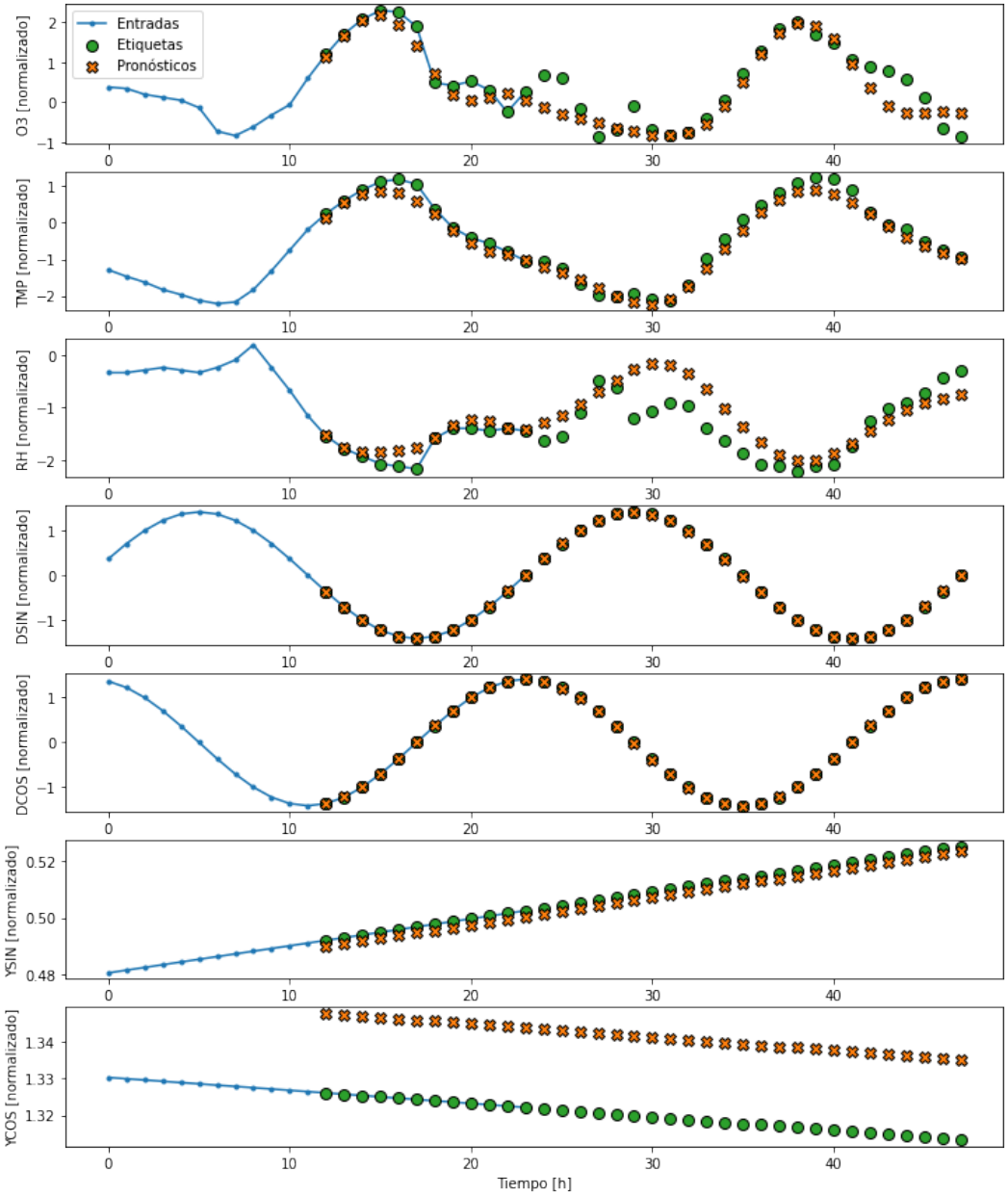

Figura 4.17: Entradas, etiquetas y pronóstico sobre muestra de la serie de tiempo multivariada de la estación Pedregal.

Método de imputación	MAE	MSE	RMSE
MTS-GAN	**0.1145**	**0.0623**	**0.2501**
KNN	0.1323	0.0700	0.2646
Valor medio	0.1312	0.0766	0.2768

Tabla 4.11: Resultados del pronóstico sobre la estación Pedregal, entrenando el modelo BiGRU con datos de la estación Merced reconstruidos con MTS-GAN, KNN y valor medio. Los valores en negrita indican los mejores resultados.

4.2.2.3. Resultados del pronóstico en la estación Xalostoc

Se utilizó el modelo BiGRU entrenado con las series de tiempo de las estación Merced para realizar el pronóstico sobre el conjunto de prueba de la estación Xalostoc.

La Figura 4.18 se muestra la predicción sobre un día de la serie de tiempo multivariada de la estación Xalostoc, donde se da como entrada las mediciones del 1 de enero del 2019 a la 1:00 hasta el 2 de Enero del 2019 a las 0:00 y se obtiene el pronóstico del 1 de Enero de 2019 a las 12:00 hasta el 2 de Enero de 2019 a las 00:00 (36 horas).

La Tabla 4.12 muestra las métricas de error sobre los resultados obtenidos en el pronóstico.

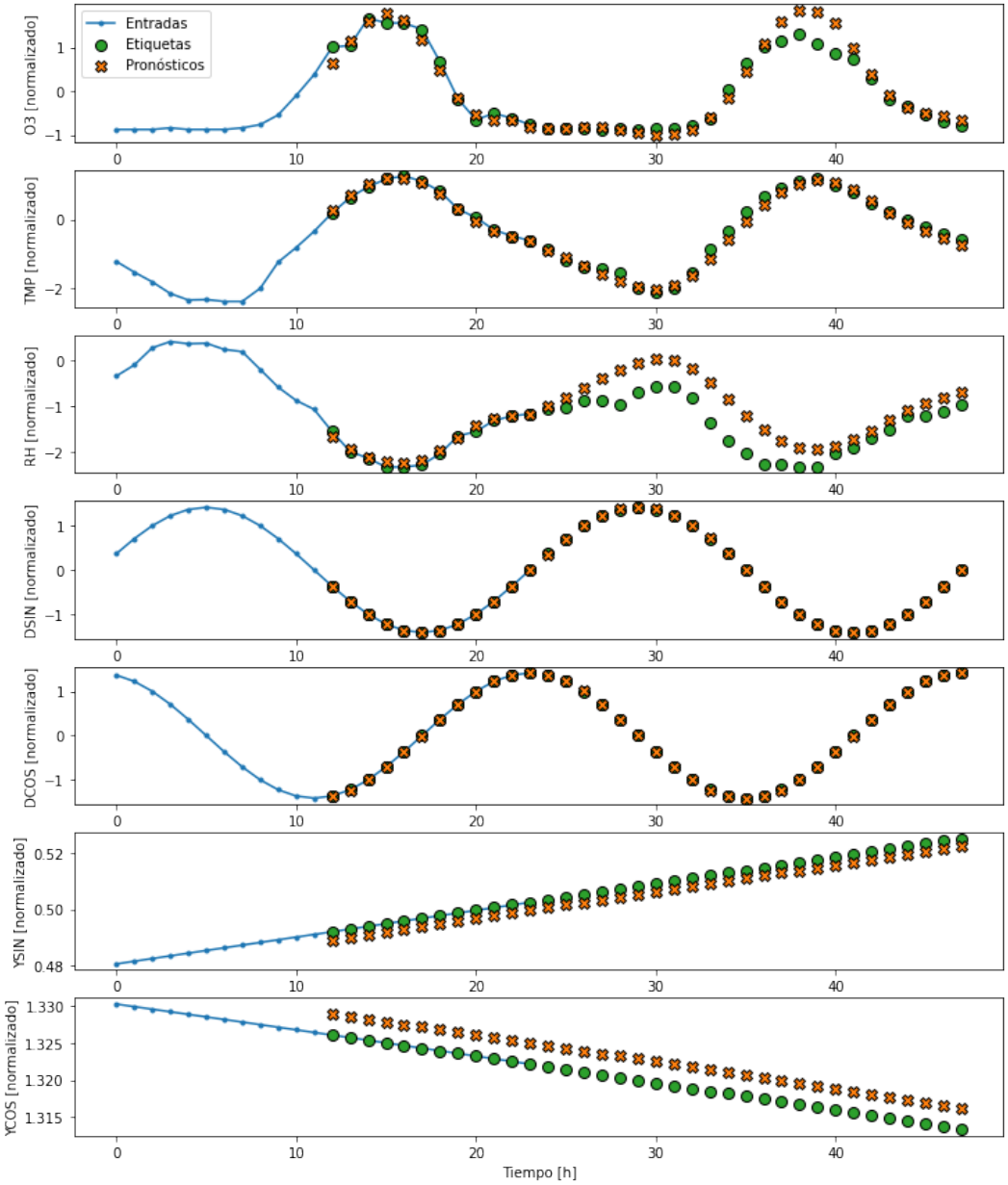

Figura 4.18: Entradas, etiquetas y pronóstico sobre muestra de la serie de tiempo multivariada de la estación Xalostoc.

Método de imputación	MAE	MSE	RMSE
MTS-GAN	**0.1012**	**0.0464**	**0.2156**
KNN	0.1111	0.0473	0.2175
Valor medio	0.1145	0.0538	0.2319

Tabla 4.12: Resultados del pronóstico sobre la estación Xalostoc, entrenando el modelo BiGRU con datos de la estación Merced reconstruidos con MTS-GAN, KNN y valor medio. Los valores en negrita indican los mejores resultados.

4.2.2.4. Resultados del pronóstico en la estación Tlalnepantla

Se utilizó el modelo BiGRU entrenado con las series de tiempo de las estación Merced para realizar el pronóstico sobre el conjunto de prueba de la estación Tlalnepantla.

La Figura 4.19 se muestra la predicción sobre un día de la serie de tiempo multivariada de la estación Tlalnepantla, donde se da como entrada las mediciones del 1 de enero del 2019 a la 1:00 hasta el 2 de Enero del 2019 a las 0:00 y se obtiene el pronóstico del 1 de Enero de 2019 a las 12:00 hasta el 2 de Enero de 2019 a las 00:00 (36 horas).

La Tabla 4.13 muestra las métricas de error sobre los resultados obtenidos en el pronóstico.

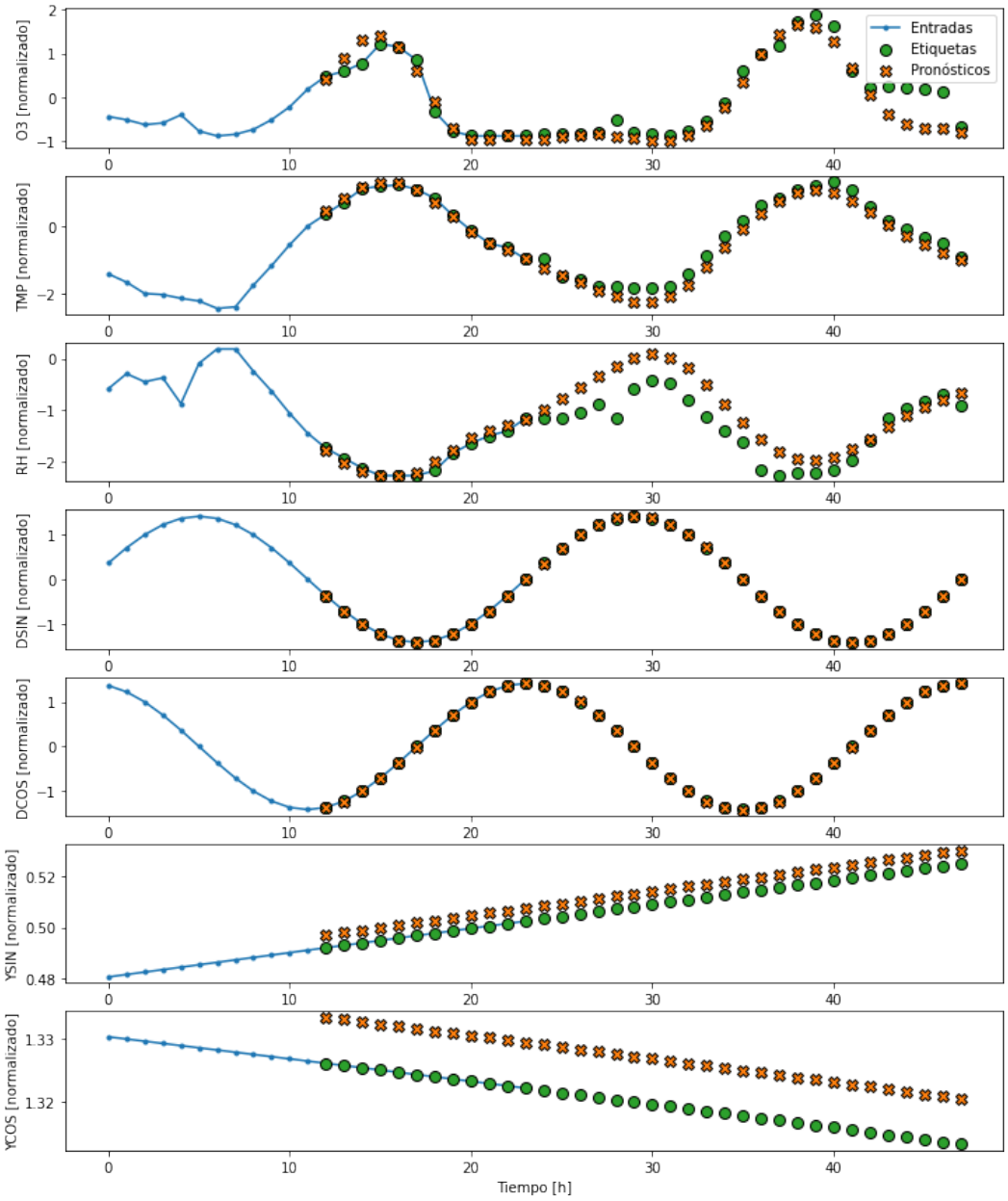

Figura 4.19: Entradas, etiquetas y pronóstico sobre muestra de la serie de tiempo multivariada de la estación Tlalnepantla.

Método de imputación	MAE	MSE	RMSE
MTS-GAN	**0.1057**	0.0539	0.2311
KNN	0.1101	**0.0473**	**0.2176**
Valor medio	0.1157	0.0562	0.2371

Tabla 4.13: Resultados del pronóstico sobre la estación Tlalnepantla, entrenando el modelo BiGRU con datos de la estación Merced reconstruidos con MTS-GAN, KNN y valor medio. Los valores en negrita indican los mejores resultados.

4.2.2.5. Resultados del pronóstico en la estación FES Acatlán

Se utilizó el modelo BiGRU entrenado con las series de tiempo de las estación Merced para realizar el pronóstico sobre el conjunto de prueba de la estación FES Acatlán.

La Figura 4.20 se muestra la predicción sobre un día de la serie de tiempo multivariada de la estación FES Acatlán, donde se da como entrada las mediciones del 1 de enero del 2019 a la 1:00 hasta el 2 de Enero del 2019 a las 0:00 y se obtiene el pronóstico del 1 de Enero de 2019 a las 12:00 hasta el 2 de Enero de 2019 a las 00:00 (36 horas).

La Tabla 4.14 muestra las métricas de error sobre los resultados obtenidos en el pronóstico.

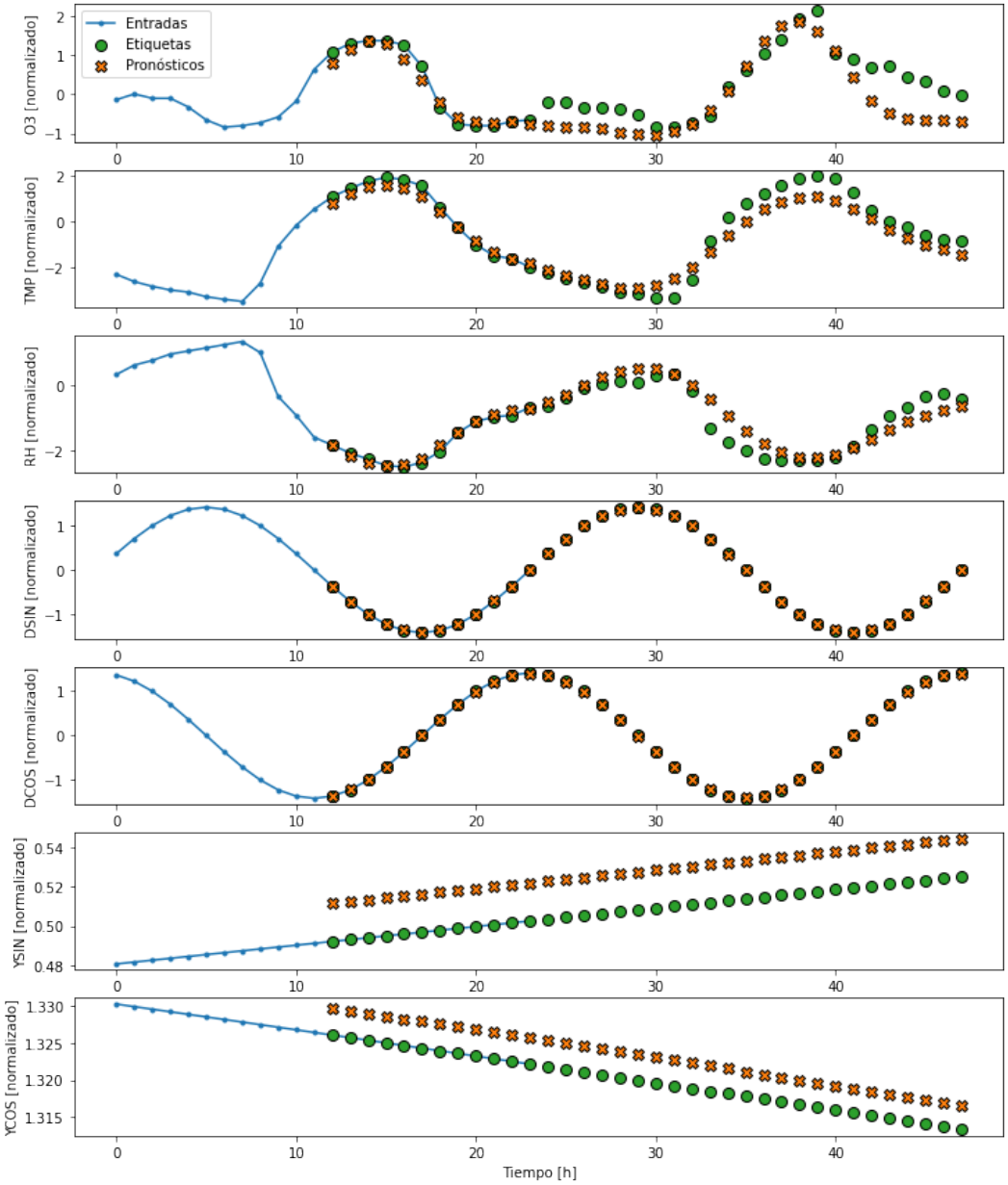

Figura 4.20: Entradas, etiquetas y pronóstico sobre muestra de la serie de tiempo multivariada de la estación FES Acatlán.

Método de imputación	MAE	MSE	RMSE
MTS-GAN	**0.1415**	0.0831	0.2880
KNN	0.1507	**0.0805**	**0.2837**
Valor medio	0.1570	0.0943	0.3071

Tabla 4.14: Resultados del pronóstico sobre la estación FES Acatlán, entrenando el modelo BiGRU con datos de la estación Merced reconstruidos con MTS-GAN, KNN y valor medio. Los valores en negrita indican los mejores resultados.

4.2.2.6. Resultados del pronóstico en la estación San Agustín

Se utilizó el modelo BiGRU entrenado con las series de tiempo de las estación Merced para realizar el pronóstico sobre el conjunto de prueba de la estación San Agustín.

La Figura 4.21 se muestra la predicción sobre un día de la serie de tiempo multivariada de la estación San Agustín, donde se da como entrada las mediciones del 1 de enero del 2019 a la 1:00 hasta el 2 de Enero del 2019 a las 0:00 y se obtiene el pronóstico del 1 de Enero de 2019 a las 12:00 hasta el 2 de Enero de 2019 a las 00:00 (36 horas).

La Tabla 4.15 muestra las métricas de error sobre los resultados obtenidos en el pronóstico.

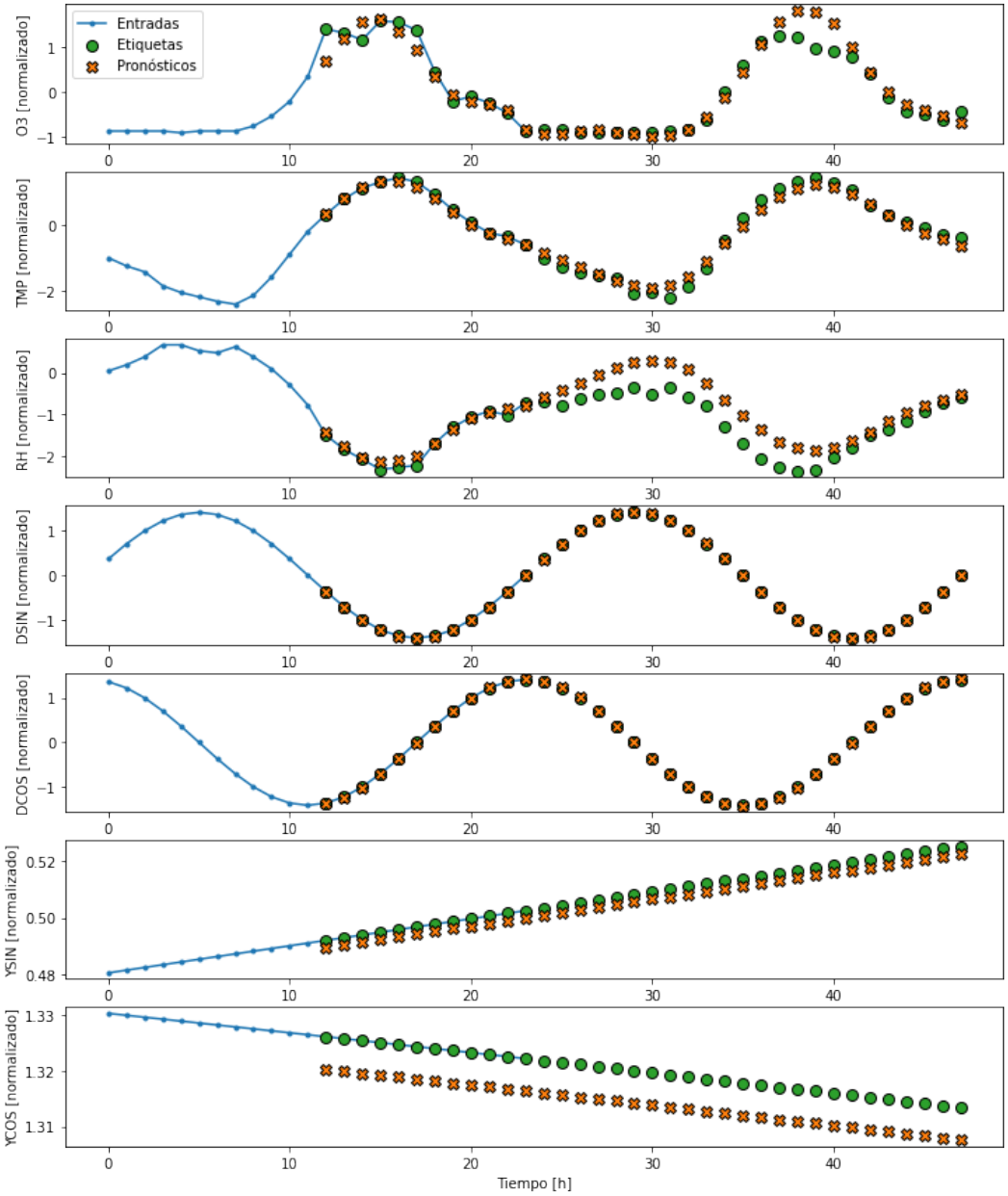

Figura 4.21: Entradas, etiquetas y pronóstico sobre muestra de la serie de tiempo multivariada de la estación San Agustín.

Método de imputación	MAE	MSE	RMSE
MTS-GAN	**0.1033**	**0.0452**	**0.2133**
KNN	0.1507	0.0805	0.2837
Valor medio	0.1236	0.0627	0.2504

Tabla 4.15: Resultados del pronóstico sobre la estación San Agustín, entrenando el modelo BiGRU con datos de la estación Merced reconstruidos con MTS-GAN, KNN y valor medio. Los valores en negrita indican los mejores resultados.

4.2.3. Análisis de resultados del pronóstico

Tal y como se esperaba, los resultados obtenidos entrenando el modelo BiGRU con las series de tiempo reconstruidas con el modelo MTS-GAN, mostraron mejores resultados sobre el conjunto de prueba de la estación Merced que cuando se reconstruyeron utilizando otros métodos, incluso en los conjuntos de prueba de la mayoría de las estaciones propuestas, cuya información no fue proporcionada en el conjunto de entrenamiento, lo que significa que el modelo entrenado con la estación seleccionada es capaz de generalizar los datos para el resto de estaciones.

En el caso de los pronósticos de año (YSIN y YCOS), puede observarse que en la mayoría de los casos existe un sesgo entre los valores reales y los valores predichos, esto en parte se debe a que al tener un periodo mucho mayor que el resto de características (p. ej. hora), el modelo tiene menos información relevante que le permita aprender completamente las características en cuestión. Para mejorar esto, se podrían eliminar aquellas instancias que no sean tan relevantes para estas características, es decir, ampliar el intervalo de muestreo, en lugar de tener una medición cada hora, realizarlo cada día. Sin embargo, esta acción no seria conveniente para el resto de características de la serie de tiempo multivariada, en donde la información que se descartaría es muy importante. Así que se optó por conservar la configuración ya establecida anteriormente, ya que, en este caso, es de mayor relevancia realizar pronósticos sobre las demás características.

La Tabla 4.16 muestra los resultados obtenidos en cada estación.

Estación	**MAE**	**MSE**	**RMSE**
Merced	0.1001	0.0462	0.2157
Pedregal	0.1145	0.0623	0.2501
Xalostoc	0.1012	0.0464	0.2156
Tlalnepantla	0.1057	0.0539	0.2311
FES Acatlán	0.1415	0.0831	0.2880
San Agustín	0.1033	0.0452	0.2133

Tabla 4.16: Resultados del pronóstico sobre distintas estaciones de la ZMVM, utilizando el modelo BiGRU, entrenado con las series de tiempo reconstruidas de la estación Merced.

Los resultados mostrados en la tabla de arriba aun pueden mejorarse utilizando algún método de optimización que permita ajustar mejor los hiperparámetros, ya que en este caso se utilizó la estrategia conocida como *grid search*, la cual es una estrategia válida y que genera buenos resultados, aunque no es capaz de focalizar la búsqueda en regiones de interés y evitar aquellas que son innecesarias.

Capítulo 5

Conclusiones

Este trabajo se enfocó principalmente en solucionar el problema de datos faltantes en las series de tiempo multivariadas con un nivel de precisión mayor que los métodos más comunes, para así reconstruir una serie de tiempo, con la cual se entrenaron modelos de redes de recurrentes que nos permitieron hacer mejores pronósticos.

Al momento de evaluar la imputación de datos sobre un conjunto de evaluación, se encontró que el modelo MTS-GAN ofrece muy buenos resultados comparados con algunos de los métodos más comunes. Los casos en los que se probaron los métodos utilizados, incluían diferentes cantidades de datos faltantes y con distintas distribuciones.

Durante el desarrollo de la experimentación, se encontró que en el primer y segundo caso, en donde los datos faltantes se presentaban de forma aleatoria y en intervalos respectivamente, el modelo MTS-GAN generó los mejores resultados para todas las estaciones en las que se probó. Esto indica que el modelo aprende del conjunto de entrenamiento, todas las relaciones que existen entre las distintas variables de las series de tiempo. El principal motivo de esta mejora, es que la arquitectura del modelo MTS-GAN toma la información de cada una de las variables de la serie a través de diferentes canales, que extraen información sobre su distribución, posteriormente se concatenan para aprender la correlación entre ellas, lo cual le permite generar datos con las distribución más cercana a la serie original. Sin embargo, para el último caso donde se probó el modelo en el cual se insertaron los datos faltantes tal y como presentaban en la serie de tiempo original, la precisión en la imputación fue menor, incluso superada por poco en algunas estaciones por el modelo KNN. Esto sucedió debido a que este caso presenta muestras con algunas variables totalmente vacías, lo que provoca que el modelo MTS-GAN no encuentre una relación con la variable que esta vacía, y por lo tanto, la representación en el espacio latente no es tan cercana a los datos originales. En cambio, si ponemos al menos algunos valores en la variable vacía anteriormente, la representación en el espacio latente es mas cercana a la distribución original de la serie. Una forma de determinar algunos de estos valores para que la representación en el espacio latente sea mejor, sería tomar los últimos valores de la muestra anterior y

los primeros de la muestra siguiente (si están disponibles) e imputarlo en la muestra incompleta, para después imputar la información restante utilizando MTS-GAN.

Los pronósticos de las series de tiempo de las estaciones propuestas mostraron mejores resultados cuando se utilizó el modelo BiGRU entrenado con la serie de tiempo reconstruida con MTS-GAN, a diferencia que cuando se reconstruyó con otros métodos. Esto indica, que el modelo MTS-GAN reconstruyó las series de tiempo con mayor precisión, y por lo tanto, la predicción es mejor. Otro punto importante, es que, el modelo BiGRU es capaz de realizar buenas predicciones sobre otras estaciones, a pesar de que este ha sido entrenado utilizando únicamente datos de una estación (Merced).

A pesar de los buenos resultados del modelo BiGRU, el rendimiento de este aún puede mejorar utilizando un método de optimización más completo (p. ej. optimización bayesiana) que nos permita encontrar el valor de parámetros más adecuado.

En general, el modelo MTS-GAN ofrece buenos resultados en todos los casos que se plantearon, incluso cuando se tenía un porcentaje alto de datos faltantes, ya sea que se entrene con el conjunto de entrenamiento de la estación que se desea imputar, cuyos resultados son mejores, o se utilice el conjunto de entrenamiento de la estación Merced, que obtuvo los segundos mejores resultados, con una notable diferencia con respecto al resto de métodos de imputación. Al momento de realizar pronósticos utilizando la serie de tiempo reconstruida con este modelo, se obtuvieron mejores resultados que cuando se reconstruyó con otros métodos, lo cual indica que se mejora la precisión del pronóstico. Por lo tanto, el modelo MTS-GAN resulta útil para reconstruir series de tiempo de contaminantes del aire en todas las estaciones, haciendo más precisos los resultados.

Como trabajo futuro, se debe considerar emplear los modelos planteados sobre series de tiempo multivariadas con un número mayor de variables, esto implica mayor consumo de recursos, pero podría mejorar aun más la precisión, al tomar en cuenta la distribución de las variables y las relaciones que existen entre ellas. Además el modelo MTS-GAN puede utilizarse con conjuntos de datos diferentes a los contaminantes del aire y datos meteorológicos, por lo tanto, se plantea utilizarlo con diferentes tipos de datos.

Bibliografía

[1] L. Bai, J. Wang, X. Ma, and H. Lu, "Air pollution forecasts: An overview," apr 2018. v, 1, 3, 12, 13, 33

[2] Z. Guo, Y. Wan, and H. Ye, "A data imputation method for multivariate time series based on generative adversarial network," *Neurocomputing*, vol. 360, no. xxxx, pp. 185–197, 2019. v, 3, 4, 10, 11, 29, 52, 68

[3] D. Vallero, *Fundamentals of Air Pollution*. Elsevier, 5 ed., 2014. 1, 33

[4] P. Belmonte Espejo and E. Gutiérrez González, "Ozono troposférico," dec 2013. 1

[5] SEDEMA, "¿Quién contamina el aire de la ZMVM?." 2

[6] M. Amann, D. Derwent, B. Forsberg, O. Hänninen, F. Hurley, M. Krzyzanowski, F. de Leeuw, S. J. Liu, C. Mandin, J. Schneider, P. Schwarze, and D. Simpson, "Health risks of ozone from long-range transboundary air pollution," *World Health Organization, Regional Office for Europe*, mar 2008. 2

[7] G. E. P. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time Series Analysis: Forecasting and Control*. No. 1, New Jersey: John Wiley Sons, 5 ed., 2015. 12

[8] R. J. Hyndman and G. Athanasopoulos, "Forecasting: Principles and Practice," 2019. 12, 33

[9] J. Mueller and L. Massaron, *Deep learning*. For Dummies, 1st ed., 2019. 13

[10] A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*. O'Reilly Media, Inc., 2nd ed., 2019. 14, 18

[11] M. Z. Alom, T. M. Taha, C. Yakopcic, S. Westberg, P. Sidike, M. S. Nasrin, M. Hasan, B. C. Van Essen, A. A. Awwal, and V. K. Asari, "A state-of-the-art survey on deep learning theory and architectures," *Electronics (Switzerland)*, vol. 8, no. 3, 2019. 14, 19, 25, 28

[12] S. O. Haykin, *Neural Networks and Learning Machines*. New York: Pearson, 3rd ed., nov 2008. 14

[13] M. A. Nielsen, "Neural Networks and Deep Learning," in *Artificial Intelligence*, pp. 39–53, determinat ed., 2015. 17

[14] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2323, 1998. 18

[15] Y. Zheng, Q. Liu, E. Chen, Y. Ge, and J. L. Zhao, "Exploiting multi-channels deep convolutional neural networks for multivariate time series classification," *Frontiers of Computer Science*, vol. 10, pp. 96–112, feb 2016. 22, 29

[16] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," tech. rep., 2013. 24

[17] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997. 25

[18] K. Cho, B.ënboer@, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation," tech. rep. 26

[19] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997. 27

[20] A. Graves, N. Jaitly, and A. R. Mohamed, "Hybrid speech recognition with Deep Bidirectional LSTM," in *2013 IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU 2013 - Proceedings*, pp. 273–278, 2013. 27

[21] V. Vukotić, C. Raymond, and G. Gravier, "A step beyond local observations with a dialog aware bidirectional GRU network for spoken language understanding," in *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, vol. 08-12-Sept, pp. 3241–3244, International Speech and Communication Association, sep 2016. 27

[22] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative Adversarial Nets," pp. 2672–2680, 2014. 27

[23] I. Goodfellow, "NIPS 2016 Tutorial: Generative Adversarial Networks," tech. rep., 2016. 27

[24] S. Pascual, A. Bonafonte, and J. Serrà, "SEGAN: Speech Enhancement Generative Adversarial Network," *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, vol. 2017-Augus, pp. 3642–3646, mar 2017. 28

[25] M. Rezaei, K. Harmuth, W. Gierke, T. Kellermeier, M. Fischer, H. Yang, and C. Meinel, "A Conditional Adversarial Network for Semantic Segmentation of Brain Tumor," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 10670 LNCS, pp. 241–252, Springer Verlag, 2018. 28

[26] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," in *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings*, International Conference on Learning Representations, ICLR, nov 2016. 29

[27] SEDEMA, "Índice AIRE y SALUD."

[28] M. Castelli, F. M. Clemente, A. Popovič, S. Silva, and L. Vanneschi, "A Machine Learning Approach to Predict Air Quality in California," *Complexity*, vol. 2020, 2020. 39

[29] T. Niu, J. Wang, H. Lu, W. Yang, and P. Du, "Developing a deep learning framework with two-stage feature selection for multivariate financial time series forecasting," *Expert Systems with Applications*, vol. 148, p. 113237, jun 2020.