

UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería



**Predicción de temperatura en centros de datos con
redes neuronales para optimización de uso de energía**

Trabajo de graduación en modalidad de Trabajo Profesional presentado
por
Douglas Raynel de León Molina
para optar al grado académico de Licenciado en Ingeniería en Ciencia de
la Computación y Tecnologías de la Información

Guatemala,
2022

UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería



**Predicción de temperatura en centros de datos con
redes neuronales para optimización de uso de energía**

Trabajo de graduación en modalidad de Trabajo Profesional presentado
por
Douglas Raynel de León Molina
para optar al grado académico de Licenciado en Ingeniería en Ciencia de
la Computación y Tecnologías de la Información

Guatemala,
2022

Vo.Bo.:

(f) 

Ing. Douglas Barrios

Tribunal Examinador:

(f) 

Ing. Douglas Barrios

(f) 

Ing. Roger Diaz

(f) × 

Ing. Eddy Castro

Fecha de aprobación: Guatemala, 7 de diciembre de 2022.

Lista de figuras	VI
Lista de cuadros	VII
Resumen	VIII
1. Introducción	1
2. Objetivos	2
2.1. Objetivo general	2
2.2. Objetivos específicos	2
3. Justificación	3
4. Metodología	4
5. Marco teórico	5
5.1. Inteligencia Artificial	5
5.2. Aprendizaje de máquina (<i>Machine Learning</i>)	5
5.3. Aprendizaje profundo (<i>Deep Learning</i>)	6
5.4. Series de tiempo	8
5.4.1. Diferencias entre modelos “tradicionales” y modelos de series de tiempo	9
5.5. Centros de datos	10
5.6. Control de temperatura en centros de datos	10
6. Marco metodológico	12
6.1. Conjunto de datos	12
6.1.1. Visualización de datos	16
6.2. Preprocesamiento	26
6.3. Modelo lineal base	38
6.3.1. Resultados	39
6.4. Modelo con capas densas	40
6.4.1. Resultados	41
6.5. Modelo de red neuronal convolucional	42
6.5.1. Resultados	43
6.6. Modelo de red neuronal recurrente	44
6.6.1. Resultados	45
6.7. Modelo autorregresivo	46

6.7.1. Resultados	46
6.8. Comparación entre modelos	47
7. Conclusiones	50
8. Recomendaciones	51
9. Bibliografía	52

5.1. Programación clásica contra <i>Machine Learning</i> .	6
5.2. Red neuronal densa básica.	7
5.3. Ejemplo de una gráfica de serie de tiempo.	8
5.4. <i>Windowing</i>	9
6.1. Distribución de sensores.	13
6.2. Señal PWM enviada desde computadora a ventiladores en el rack.	13
6.3. Medición de temperatura para el caso 1.	14
6.4. Medición de temperatura para el caso 2.	14
6.5. Medición de temperatura para el caso 3.	15
6.6. Medición de temperatura para el caso 4.	15
6.7. Medición de temperatura para el caso 5.	16
6.8. Velocidad de ventilador de entrada para el caso 1.	17
6.9. Velocidad de ventilador de salida para el caso 1.	17
6.10. Media de temperatura de entrada para el caso 1.	18
6.11. Media de temperatura de salida para el caso 1.	18
6.12. Velocidad de ventilador de entrada para el caso 2.	19
6.13. Velocidad de ventilador de salida para el caso 2.	19
6.14. Media de temperatura de entrada para el caso 2.	20
6.15. Media de temperatura de salida para el caso 2.	20
6.16. Velocidad de ventilador de entrada para el caso 3.	21
6.17. Velocidad de ventilador de salida para el caso 3.	21
6.18. Media de temperatura de entrada para el caso 3.	22
6.19. Media de temperatura de salida para el caso 3.	22
6.20. Velocidad de ventilador de entrada para el caso 4.	23
6.21. Velocidad de ventilador de salida para el caso 4.	23
6.22. Media de temperatura de entrada para el caso 4.	24
6.23. Media de temperatura de salida para el caso 4.	24
6.24. Velocidad de ventilador de entrada para el caso 5.	25
6.25. Velocidad de ventilador de salida para el caso 5.	25
6.26. Media de temperatura de entrada para el caso 5.	26
6.27. Media de temperatura de salida para el caso 5.	26
6.28. Velocidad de ventilador de entrada normalizada para el caso 1.	27
6.29. Velocidad de ventilador de salida normalizada para el caso 1.	28
6.30. Media de temperatura de entrada normalizada para el caso 1.	28
6.31. Media de temperatura de salida normalizada para el caso 1.	29
6.32. Velocidad de ventilador de entrada normalizada para el caso 2.	29

6.33. Velocidad de ventilador de salida normalizada para el caso 2.	30
6.34. Media de temperatura de entrada normalizada para el caso 2.	30
6.35. Media de temperatura de salida normalizada para el caso 2.	31
6.36. Velocidad de ventilador de entrada normalizada para el caso 3.	31
6.37. Velocidad de ventilador de salida normalizada para el caso 3.	32
6.38. Media de temperatura de entrada normalizada para el caso 3.	32
6.39. Media de temperatura de salida normalizada para el caso 3.	33
6.40. Velocidad de ventilador de entrada normalizada para el caso 4.	33
6.41. Velocidad de ventilador de salida normalizada para el caso 4.	34
6.42. Media de temperatura de entrada normalizada para el caso 4.	34
6.43. Media de temperatura de salida normalizada para el caso 4.	35
6.44. Velocidad de ventilador de entrada normalizada para el caso 5.	35
6.45. Velocidad de ventilador de salida normalizada para el caso 5.	36
6.46. Media de temperatura de entrada normalizada para el caso 5.	36
6.47. Media de temperatura de salida normalizada para el caso 5.	37
6.48. Ventana de temperatura de salida en posición 60.	38
6.49. Modelo lineal base	38
6.50. Predicciones de modelo lineal.	39
6.51. Modelo ANN	41
6.52. Predicciones de modelo con capas densas.	42
6.53. Modelo CNN	43
6.54. Predicciones de modelo convolucional.	44
6.55. RNN con capa LSTM.	45
6.56. Predicciones de modelo RNN con capa LSTM.	46
6.57. Predicciones de modelo RNN autorregresivo.	47
6.58. Comparación entre resultados de modelos	48
6.59. Comparación entre tiempo de entrenamiento de modelos	49

Lista de cuadros

6.1. Distribución de observaciones por caso.	12
6.2. Tiempo de entrenamiento por caso para el modelo lineal.	39
6.3. Resultados de modelo lineal.	39
6.4. Tiempo de entrenamiento por caso para el modelo con capas densas.	42
6.5. Resultados de modelo con capas densas.	42
6.6. Tiempo de entrenamiento por caso para el modelo convolucional.	43
6.7. Resultados de modelo convolucional.	44
6.8. Tiempo de entrenamiento por caso para el modelo RNN.	45
6.9. Resultados de modelo RNN con capa LSTM.	45
6.10. Tiempo de entrenamiento por caso para el modelo con capas densas.	46
6.11. Resultados de modelo RNN autorregresivo	47

El siguiente trabajo se enfoca en la elaboración, evaluación y comparación de diferentes modelos para predicción de temperatura en centros de datos. Específicamente, se enfoca en la predicción de temperatura a nivel de rack. Esto se logra a través de la implementación de técnicas de visualización y preprocesamiento de datos. Sumado a esto, se implementan técnicas de predicción para series de tiempo. Este trabajo forma parte de un proyecto realizado en conjunto con la universidad Yang Ming Chiao Tung de Taiwán, en el cual se tiene como objetivo la optimización de energía en el enfriamiento de centros de datos. Este trabajo representa el módulo de predicción de temperatura para luego poder alimentar estas predicciones a un controlador de los ventiladores.

Se realizó un modelo lineal base, un modelo de red neuronal con capas densas, un modelo de red neuronal convolucional, uno de red neuronal recurrente de tipo LSTM y uno autorregresivo recurrente. Posterior al desarrollo de los mismos, se evaluaron y compararon con la raíz del error cuadrático medio como principal métrica. Sumado esto, se comparó el tiempo de entrenamiento de los mismos.

El control de temperatura es una parte importante para la seguridad y funcionamiento de los centros de datos. Debido a esto, es una práctica común el utilizar el sistema de enfriamiento en valores constantes para evitar accidentes y pérdidas de servicio debido al sobrecalentamiento de las máquinas. Esto presenta un problema debido a que en ocasiones se utiliza más energía de la necesaria para mantener los centros de datos seguros y en funcionamiento, lo que presenta una oportunidad de optimización. Una de las partes importantes que involucra este problema es lograr predecir el ambiente térmico de los centros de datos [1] [2].

El objetivo de este trabajo es diseñar, desarrollar y evaluar modelos de redes neuronales para la predicción de la temperatura en centros de datos. Esto se llevará a cabo utilizando técnicas de predicción para series de tiempo, inteligencia artificial y ciencia de datos para el preprocesamiento, visualización de datos y desarrollo de los modelos. Se desarrollará un modelo base y modelos modificados para poder evaluar y comparar el rendimiento de los mismos. Además, se realizará una investigación de todos los temas técnicos necesarios para la realización de los modelos como lo son los centros de datos, el ambiente térmico de los mismos, las series de tiempo, inteligencia artificial y ciencia de datos.

Este proyecto forma parte de un proyecto realizado en conjunto con la Universidad Yang Ming Chiao Tung de Taiwán, que involucra un controlador que se alimenta con las predicciones de los modelos realizados en este trabajo. Esto con el objetivo de poder controlar la temperatura dentro de los centros de datos por medio de las modificaciones a la velocidad de los ventiladores realizadas por el controlador. Para este tipo de problemas, dependiendo del enfoque, se pueden tomar diferentes acercamientos: a nivel de instalación, a nivel de rack, o incluso más granular. En este trabajo se decidió trabajar la predicción de temperatura a nivel de rack.

2.1. Objetivo general

Desarrollar y evaluar modelos de predicción de temperatura con redes neuronales para el control de temperatura en centros de datos con el fin de modelar los cambios de temperatura en un rack.

2.2. Objetivos específicos

- Investigar técnicas para control de temperatura en centros de datos.
- Implementar técnicas de predicción para series de tiempo.
- Implementar técnicas de visualización de datos.
- Elaborar un modelo de redes neuronales base.
- Elaborar un modelo de redes neuronales convolucionales.
- Elaborar un modelo de redes neuronales recurrente.
- Evaluar y comparar los modelos.

Los centros de datos son un tipo de infraestructura tecnológica que consiste de equipo de procesamiento, equipo de almacenamiento de datos y una red de interconexión. Este tipo de instalaciones ha tenido un auge muy importante en los últimos años con la tendencia de infraestructura como servicio (IaaS por sus siglas en inglés) y plataformas como Google Cloud, Amazon Web Services, Digital Ocean, entre otros. Con este modelo de negocio y con los centros de datos locales el proveedor de la infraestructura se encarga de la seguridad del equipo, lo que incluye el control de temperatura del mismo [3].

Uno de los problemas más complejos que involucra el mantenimiento de un centro de datos es el control de la temperatura del equipo. Los centros de datos diseñados y construidos en los últimos 10 años son capaces de enfriar hasta 3KW de carga de calor por gabineta. Esto normalmente involucra un diseño de plenos con suelo levantado (*raised floor plenum*) con altura de 18 a 24 pulgadas, techos con alturas de 8 a 9 pies y unidades CRAC (*Computer Room Air Conditioning* distribuidas en la habitación [4]. En este son las unidades CRAC las que se encargan del mantenimiento de la temperatura. Sin embargo, debido a la compleja naturaleza del ambiente térmico del equipo causado por la demanda de cómputo, es común que el equipo de enfriamiento este configurado con valores por defecto. Esto implica un uso de energía constante por el equipo de enfriamiento en todo momento, lo que presenta una oportunidad de optimización.

Una de las propuestas para la predicción de temperatura en los centros de datos es el uso de aprendizaje de máquina e inteligencia artificial para la implementación de un modelo o agente que nos ayude a modelar de forma más precisa el ambiente térmico. La técnica propuesta en este trabajo es la implementación de diferentes redes neuronales entrenadas con un conjunto de datos proveído por la Universidad Nacional Yang Ming Chiao Tung de Taiwán. El objetivo de estas redes es poder modelar el ambiente término a nivel de rack, para así poder hacer predicciones de la temperatura y poder alimentarlas a un controlador para modificar la velocidad de los ventiladores en el rack.

Para la realización del proyecto se utilizaron un conjunto de datos proveído por la Universidad Nacional Yang Ming Chiao Tung de Taiwán. Este conjunto de datos provee 5 casos de mediciones de velocidad de ventilador de entrada, temperaturas de entrada, velocidad de ventilador de salida y temperaturas de salida. Sumado a esto, para la investigación de los temas técnicos, se hizo una revisión de literatura en relación al tema de predicción de temperaturas en centros de datos. Esto se complementó con la investigación de los temas adicionales como lo son la inteligencia artificial, aprendizaje de máquina, aprendizaje profundo, las series de tiempo, entre otros.

Para la realización de los modelos de predicción, se utilizó el lenguaje de programación Python en conjunto con librerías populares para inteligencia artificial y ciencia de datos como lo son Tensorflow, Matplotlib y Numpy. Se realizó una visualización inicial del conjunto de datos para los diferentes casos que contiene. Posteriormente, se llevó a cabo el preprocesamiento del mismo, seleccionando las variables significativas para la realización de los modelos. Luego, se creó un modelo de redes neuronales lineal base, uno de redes neuronales convolucionales y uno de redes neuronales recurrentes. Es importante mencionar que todos los modelos requieren de cierto preprocesamiento para dividir la data en conjuntos de datos de entrenamiento, validación y prueba y también para el *windowing* que requiere la metodología de series de tiempo. Se evaluaron los modelos con raíz cuadrada del error cuadrático. Es importante mencionar que este trabajo se enfoca en el modelado para la predicción de temperatura con el objetivo de alimentar un controlador para manejar el sistema de enfriamiento en un centro de datos.

5.1. Inteligencia Artificial

Según Russell y Norvig [5], el campo de inteligencia artificial, o AI por sus siglas en inglés, se interesa por el entendimiento y la construcción de la inteligencia y los entes inteligentes. En otras palabras, lo que nosotros entendemos como inteligencia. Esto puede englobar el entendimiento, la percepción, la predicción y la manipulación de un mundo que es mucho más grande y complicado que nosotros mismos. En términos prácticos, este campo se enfoca en crear máquinas que puedan computar de forma efectiva en una variedad de situaciones. Cabe destacar que la inteligencia artificial es un campo relativamente nuevo, en el cual aún hay mucho camino por recorrer.

Históricamente, se ha descrito la inteligencia artificial de muchas formas. Algunas de las definiciones la describen en términos de la fidelidad con el rendimiento humano, mientras que otras prefieren definirla de forma más abstracta con la racionalidad, que hace referencia a hacer lo correcto. Además, algunos consideran la inteligencia como una propiedad de procesos de pensamiento internos y razonamiento, mientras otros se enfocan en el comportamiento. De esta forma, Russell y Norvig definen dos dimensiones: humano-racional y pensamiento-comportamiento con 4 posibles combinaciones, las cuales han tenido bastante campo en investigación y se han ayudado unas entre otras. Entre estos enfoques se acumulan aportes de diferentes disciplinas como la filosofía, matemáticas, economía, neurociencia, psicología, computación, lingüística, entre otras.

5.2. Aprendizaje de máquina (*Machine Learning*)

Russell y Norvig [5] describen el aprendizaje de un agente como una mejora en el rendimiento luego de haber hecho observaciones en su universo. Cuando el agente es una computadora, se le llama aprendizaje de máquina, o *Machine Learning* en inglés. Este es un cambio de paradigma bastante importante, puesto que tradicionalmente el modelo o programa es diseñado e implementado por una persona. En el aprendizaje de máquina, es la misma máquina la que observa la data, construye el modelo con base a esas observaciones y utiliza el modelo generado. Esto es importante, puesto que en muchos problemas los humanos no somos capaces de anticiparnos a todas las posibles situaciones futuras o incluso no somos capaces de programar una solución para dado problema. La diferencia entre la programación clásica y el aprendizaje de máquina se puede apreciar en la Figura 5.1. A

pesar de que el aprendizaje de máquinas es un subconjunto de la inteligencia artificial, es un campo bastante amplio por sí mismo. Este se utiliza para muchos tipos de problemas, como puede ser la regresión, la segmentación, clasificación, entre otros. Para estos problemas se utilizan diferentes tipos de modelos como los árboles de decisión, modelos lineales, modelos de segmentación de datos como K-medias, modelos de ensamblado como *random forests*, entre otros.

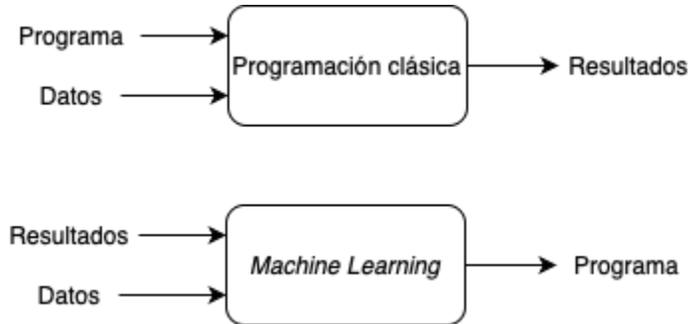


Figura 5.1: Programación clásica contra *Machine Learning*.

Entre estos modelos existen clasificaciones dependiendo de la retroalimentación que tiene el modelo dependiendo de las entradas: el aprendizaje supervisado, el no supervisado y el aprendizaje reforzado. En el aprendizaje supervisado, el agente observa los pares de entrada-salida y aprende una función que mapea de la entrada a la salida. Un ejemplo de aprendizaje supervisado es la predicción de temperatura en centros de datos, en el que el modelo trata de aprender la función que mejor predice la salida dada cierta entrada. En el aprendizaje no supervisado, el agente aprende de los patrones de la entrada sin ninguna retroalimentación explícita. El ejemplo más claro de este tipo de aprendizaje es la segmentación (o *clustering* en inglés), en el cual la máquina aprende a separar las observaciones en grupos, dependiendo de qué tanto pertenecen al mismo sin ninguna etiqueta dada por el diseñador. En el aprendizaje reforzado, el agente aprende de premios y penalizaciones. El ejemplo clásico de este es la máquina que aprende a jugar ajedrez, como AlphaZero.

Para los proyectos de *Machine Learning* existe cierta metodología que se ha ido formando a través de los años y que aún está evolucionando. Todo empieza con la formulación del problema, como puede ser “la predicción de temperatura en un centro de datos”. Luego, debemos realizar una recolección de datos. Dependiendo del conjunto de datos que obtengamos, puede ser necesario hacer un aumento o balanceo del mismo. En este punto también es importante tomar en cuenta los datos atípicos en nuestra data, puesto que puede llegar a tener efectos no deseados en el modelo dado que no son representativos. Luego, se pasa al *feature engineering*, en donde nos aseguramos que nuestra data sea fácil de procesar para nuestro modelo, como lo puede ser un *one-hot encoding*. En este paso es donde generamos el *windowing* y también normalizamos nuestra data. Además, se debe de hacer un análisis exploratorio de datos, en el que se evalúa nuestro conjunto de datos de forma cuantitativa y visual. Posteriormente, se selecciona el modelo que consideremos que mejor se adapte a nuestro problema y se entrena con nuestra data. Este paso también debe incluir la evaluación de los modelos utilizando diferentes tipos de métricas como lo pueden ser la precisión, el puntaje f-1 (*f-1 score*), ROC, error cuadrático medio entre otras. Por último, se debe de poner el modelo seleccionado en operación, monitorearlo y mantenerlo.

5.3. Aprendizaje profundo (*Deep Learning*)

Según IBM [6], *Deep Learning* es un subconjunto de *Machine Learning*, en que se aplican redes neuronales de tres o más capas. Estas redes neuronales tratan de imitar el comportamiento del cerebro y la neuronas (aunque estén muy lejos de lograrlo). El *Deep Learning* está aplicado en

muchas aplicaciones y servicios de inteligencia artificial que automatizan tareas de forma que no se necesite intervención humana, como asistentes virtuales, detectores de fraude y campos nuevos como los carros que se manejan solos. Estas redes intentan imitar el cerebro humano con entradas, pesos y sesgos.

Las redes neuronales artificiales (*artificial neural networks* o ANNs) consisten de múltiples capas de nodos interconectados que construyen sobre las capas anteriores para refinar y optimizar las predicciones o clasificaciones, como se puede visualizar en la Figura 5.2. Esta progresión de la computación a través de las capas se le conoce como *forward propagation*. Las capas de entrada y de salida se les conoce como capas visibles. Otro aspecto importante a mencionar en las redes neuronales es el *backpropagation*, la forma a través de la cual aprenden estas redes. Este proceso utiliza algoritmos como el descenso gradiente para ir de la capa de salida hacia atrás en la red, ajustando los pesos dependiendo del error encontrado en la salida. El *forward propagation* y *backpropagation*, en combinación, permiten a las redes neuronales artificiales hacer predicciones y corregir dependiendo del error y de esa forma aprender.

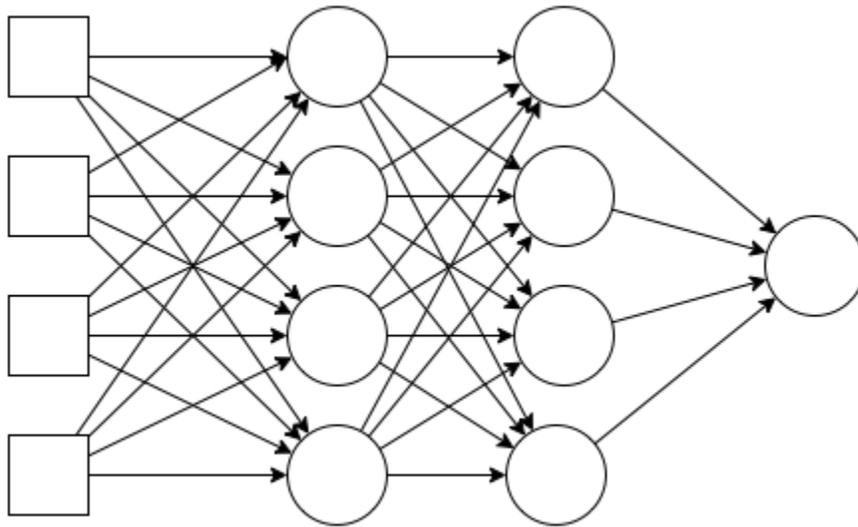


Figura 5.2: Red neuronal densa básica.

Se puede apreciar que tiene una capa de entrada con 4 nodos y una de salida con un solo nodo.

Además, dependiendo del tipo de data y el caso de uso, existe una variedad de tipos de redes neuronales aparte de las mencionadas anteriormente que mejor se puede adaptar al problema dado. Otro tipo de red neuronal son las redes neuronales convolucionales (*convolutional neural networks* o CNNs en inglés). Este tipo de redes se utilizan comúnmente para problemas en los que se tiene localidad espacial o adyacencia en los datos, como lo pueden ser imágenes. Este tipo de redes involucran al menos una capa convolucional en la que se aplica un kernel, que hace referencia a un patrón de pesos, a múltiples regiones locales. Este tipo de redes también puede involucrar capas de *pooling*, en las que se resume un conjunto de unidades de la capa anterior a un solo valor [5]. Este tipo de redes puede ser útiles para el problema de este trabajo puesto que las observaciones presentan adyacencia temporal debido a la naturaleza de las series de tiempo. Específicamente, las capas convolucionales de una sola dimensión se prestan para problemas de series de tiempo.

Otro tipo de red neuronal que tiene mucho potencial para el problema de este trabajo son las redes neuronales recurrentes (*recurrent neural networks* o RNNs en inglés). Tensorflow [7], describe a este tipo de redes neuronales como “poderosas para el modelado de data secuencial como las series de tiempo o lenguaje natural”. En este tipo de redes se permiten ciclos en el grafo computacional, lo

que significa que un ciclo puede tener como entrada la salida del ciclo anterior. Esto permite que la red tenga “memoria”, o que los ciclos anteriores afecten la respuesta de la red en el ciclo actual. Por este motivo las RNNs son una herramienta con mucho potencial para analizar data secuencial [5]. Una de las arquitecturas de RNN más populares son las *long short-term memory* o LSTM. Este tipo de arquitectura evita problemas que previenen a otras RNNs entrenarse y escalar, que son problemas comunes en este tipo de redes. Los componentes esenciales de este tipo de redes son el *memory cell* y las *gating units*. Algunos de las aplicaciones interesantes de este tipo de redes son la traducción de texto y la generación de escritura [8].

5.4. Series de tiempo

Hacer un pronóstico es utilizar toda la información disponible (data histórica y nuestro propio conocimiento) para predecir el futuro de la forma más precisa posible. Un pronóstico cuantitativo se puede realizar cuando tenemos información numérica del pasado y podemos asumir que algunos aspectos de los patrones del pasado continuarán en el futuro. La mayoría de los problemas de predicción cuantitativa utilizan data de serie de tiempo o datos transversales [9].

Todo lo que sea observado de forma secuencial en el tiempo se puede considerar una serie de tiempo. Por ejemplo, la temperatura del aire de entrada y salida en un centro de datos. Para este tipo de problemas, el objetivo es estimar cómo va a continuar la secuencia de datos en el futuro. La Figura 5.3 muestra un ejemplo de una gráfica de una serie de tiempo.

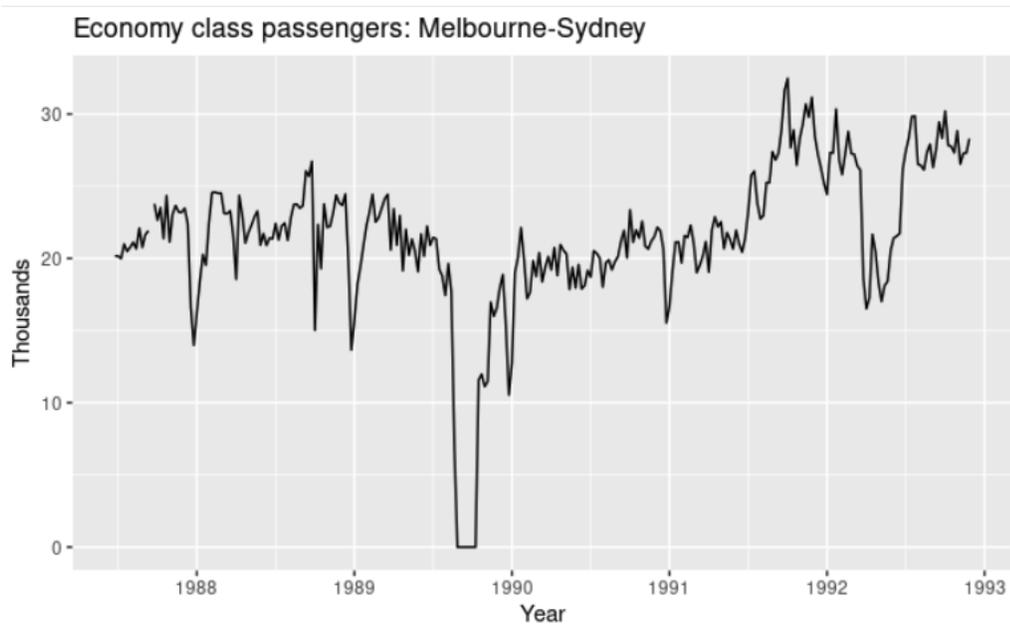


Figura 5.3: Ejemplo de una gráfica de serie de tiempo.

Carga de pasajeros semanales en Ansett Airlines. Tomado de *Forecasting: Principles and Practica*, por Hundman R.J. y Athanasopoulos G., 2018, recuperado de OTexts.com/fpp2.

5.4.1. Diferencias entre modelos “tradicionales” y modelos de series de tiempo

En un modelo tradicional, uno de los pasos estándar es dividir los datos en conjuntos de entrenamiento, validación y de prueba. Una de las prácticas comunes para este paso es seleccionar los datos para los diferentes conjuntos de forma aleatoria. Sin embargo, debido a que en un problema de serie de tiempo nuestra data es secuencial, no podemos hacer esa selección al azar. Es por esto que se debe seleccionar estos conjuntos de forma adyacente para la creación del modelo. Otro aspecto que cambia al crear modelos de series de tiempo es la normalización de los datos. En un modelo tradicional de aprendizaje de máquina, es común utilizar la media y desviación estándar sobre todo el conjunto de datos para este proceso. Sin embargo, en un modelo de series de tiempo nuestro modelo no debería de tener acceso a datos en el futuro (implícitos en la media y la desviación estándar) en dado momento. Esto significa que se debe normalizar con una media móvil [10] o buscar algún otro tipo de normalización.

Windowing

Otro de los aspectos importantes de crear un modelo de serie de tiempo es el *windowing*. Esto debido a que la forma de la data secuencial no es la adecuada para el entrenamiento de un modelo. En la Figura 6.47 se muestra un ejemplo simple en el cual podemos apreciar la comparación entre la data de serie de tiempo y esta misma data preprocesada luego del *windowing*. Este preprocesamiento de los datos permite convertir la data secuencia a un conjunto de datos que se puede dividir en variables dependientes e independientes. En el ejemplo, podemos ver que y depende de X , el paso anterior en el tiempo.

a)	t	Temperatura
	1	27
	2	28
	3	29
	4	30
	5	31

b)	X	y
	?	27
	27	28
	28	29
	29	30
	30	31
	31	?

Figura 5.4: *Windowing*

a) Data secuencial sin preprocesamiento. b) Data preprocesada con *windowing* para permitir entrenamiento de modelo.

Los modelos hacen predicciones con base a una ventana de muestras consecutivas de datos. Estas ventanas deben de incluir: el ancho, que hace referencia a la cantidad de pasos en el tiempo de las ventanas de entrada y etiquetas; el offset de tiempo entre ellas; y cuales características son utilizadas como entradas, etiquetas o ambas. La ventana es muy importante para este tipo de problemas puesto que determina si nuestro modelo puede hacer predicciones de una o múltiples salidas y también si nuestro modelo puede hacer predicciones en uno o múltiples pasos en el tiempo. En el caso del problema que trata este proyecto, la meta es lograr construir un modelo que prediga múltiples pasos en el tiempo y logre predecir las 135 temperaturas de salida. Para este tipo de modelos hay dos acercamientos: la predicciones “single shot”, en las cuales el modelo hace la predicciones de la secuencia de datos entera en un solo paso; y las predicciones autorregresivas, en las cuales las salidas

del modelo son alimentadas de regreso al modelo en cada paso y las predicciones son condicionadas en las anteriores [10].

5.5. Centros de datos

Según Cisco [11], en su forma más simple, un centro de datos es una instalación que las organizaciones utilizan para alojar sus aplicaciones y datos críticos. Estos están diseñados como una red de recursos de computación y almacenamiento que permiten compartir aplicaciones y datos. Los componentes principales de los centros de datos incluyen: *routers*, *switches*, *firewalls*, sistemas de almacenamiento, servidores, entre otros. Los centros de datos modernos han tendido a un cambio de servidores *on-premise* tradicionales a redes virtuales en la nube. En la actualidad existen más de 4,900 localidades de centros de datos distribuidos en 129 países alrededor del mundo [12].

Los centros de datos son muy importantes para el mundo empresarial de TI. Esto debido a que soportan aplicaciones y actividades del negocio como el correo electrónico e intercambio de archivos, aplicaciones de productividad, manejo de relaciones con clientes (CRMs), planificación de recursos empresariales (ERPs) y bases de datos, *Big Data*, inteligencia artificial y aprendizaje de máquina, servicios de comunicación y colaboración, entre otros. Es por esto que los centros de datos son considerados como recursos críticos para las empresas que los utilizan.

5.6. Control de temperatura en centros de datos

Un diseño convencional de centro de datos es el *Raised Floor Plenum* (RFP), en el que la unidad CRAC (*Computer Room Air-Conditioning*) lleva el aire frío a los racks de servidor a través de perforaciones en el pleno. Los racks de servidor son colocados en filas, formando pasillos fríos y calientes. El aire frío pasa a través de los servidores y el aire caliente es expulsado a través del techo de los pasillos calientes.

Sin embargo, debido a la naturaleza de las operaciones en un centro de datos, la carga computacional es altamente dinámica y, por lo tanto, también la generación de calor de los diferentes racks en los mismos. En ausencia de algoritmos basados en física, los sistemas de acondicionamiento de aire son típicamente operados en puntos predeterminados [1]. Esto resulta en un consumo de electricidad subóptimo.

Sumado a esto, los modelos computacionales de dinámica de fluidos (*Computational Fluid Dynamics* o CFD en inglés) son muy costosos computacionalmente para ser utilizados en algoritmos de control. Esto es importante puesto que los centros de datos pueden llegar a consistir de cientos de servidores en rack, unidades de enfriamiento y sistemas de backup y la demanda de los mismos ha incrementado. Esto significa que el problema es importante puesto que tiene implicaciones a nivel de costos de la operación de los mismos y también a nivel ambiental. Es por esto que es valiosa la predicción de temperatura con modelos basados en datos [1].

Según Fulpagare et al. [1], de las variables en un centro de datos, las más importantes incluyen: las temperaturas de aire entrada y salida en los racks de servidor, la velocidad del ventilador del servidor, temperaturas de CPU y temperatura de las unidades CRAC. Se menciona también que algunos estudios han utilizado herramientas CFD de estado estable para entender el efecto de estas variables. Otros estudios se han enfocado en el modelado de estado estable enfocados en el nivel de rack con diferentes motivos y mejoras. Sin embargo, debido al tiempo y a los recursos computacionales requeridos para una simulación CFD detallada, no es un método práctico para utilizar en controladores en tiempo real.

Saiyad et al. [3] mencionan que se ha realizado análisis CFD para entender el flujo de data de diferentes parámetros como las temperaturas de entrada y salida de los racks. Se han desarrollado modelos basados en física para predecir la utilización de energía. Fulpagare et al. realizaron modelos lineales multivariados de predicción basados en datos a nivel de rack [1] y a nivel de instalación [2] entrenados con data de CFD y experimental. También se han realizado modelos predictivos de *proper orthogonal decomposition* (POD). Sin embargo, no son aplicables en casos reales debido a bajos resultados de extrapolación para casos con múltiples entradas y salidas. Recientemente, se han explorado las redes neuronales artificiales (ANNs, por sus siglas en inglés) para capturar la no-linealidad de los sistemas en centros de datos. Saiyad et al. [3], realizaron un modelo de ANN aplicable a diferentes configuraciones de centros de datos entrenado con datos de simulaciones CFD y datos experimentales a nivel de rack y de instalación.

6.1. Conjunto de datos

El conjunto de datos fue proveído por la Universidad Yang Ming Chiao Tung de Taiwán. Este consiste en 1753 observaciones divididas en 5 casos independientes. Estos contienen las siguientes variables de entrada y de salida: tiempo (s), velocidad de ventilador (PWM) y 135 mediciones de temperatura (C°). Esto da un total de 272 variables (el tiempo es excluido, puesto que se utiliza como índice). Estos datos fueron capturados con un *data logger* para registrar los datos capturados por un *thermocouple* que se encarga de hacer las mediciones de temperatura. Se tomaron en cuenta estas variables puesto que son las que generalmente son monitoreadas en un centro de datos. En la Tabla 6.1 se presenta la distribución de observaciones según el caso. Para facilitar las pruebas de los modelos se decidió utilizar el primer caso exclusivamente para pruebas, lo que equivale a 181 observaciones (10.45% del total).

Caso	No. observaciones	Porcentaje
Caso 1	183	10.45%
Caso 2	204	11.64%
Caso 3	211	12.04%
Caso 4	371	21.18%
Caso 5	783	44.69%

Tabla 6.1: Distribución de observaciones por caso.

A continuación en la Figura 6.1 se muestra la distribución de sensores al momento de realizar las mediciones de temperatura. De igual forma en la Figura 6.2 se muestra la distribución de los ventiladores para la medición de la variable de velocidad de ventilador.

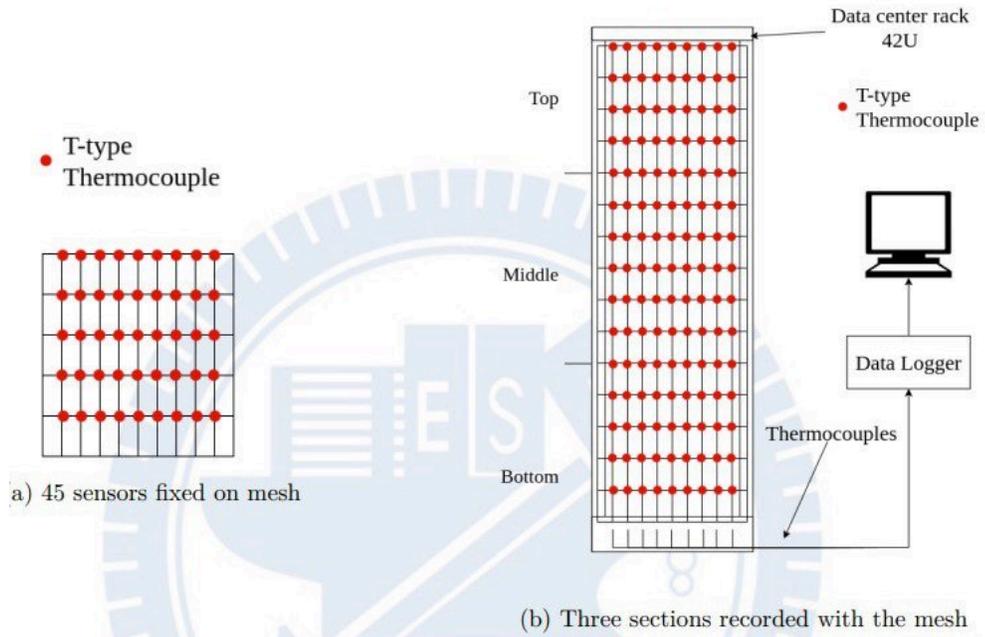


Figura 6.1: Distribución de sensores.

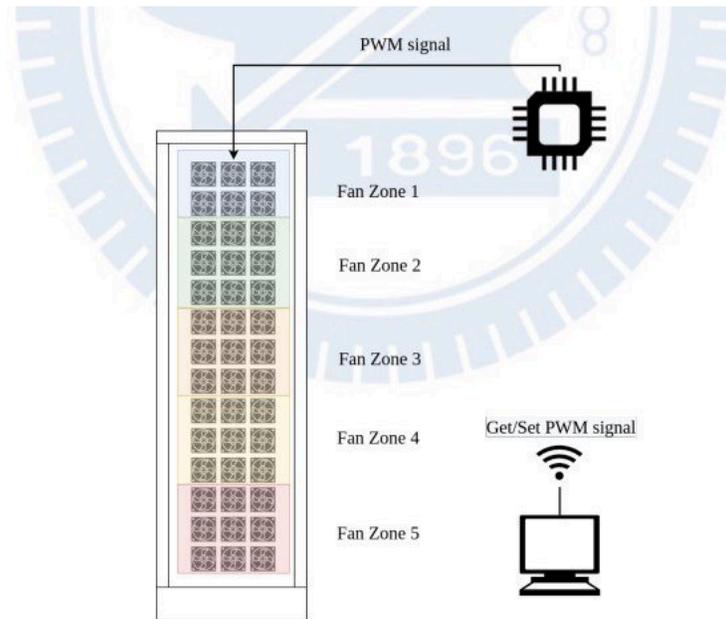


Figura 6.2: Señal PWM enviada desde computadora a ventiladores en el rack.

De igual forma, a continuación se muestran mediciones de temperatura de entrada y de salida para los 5 casos. Estas figuras representan una medición en el tiempo.

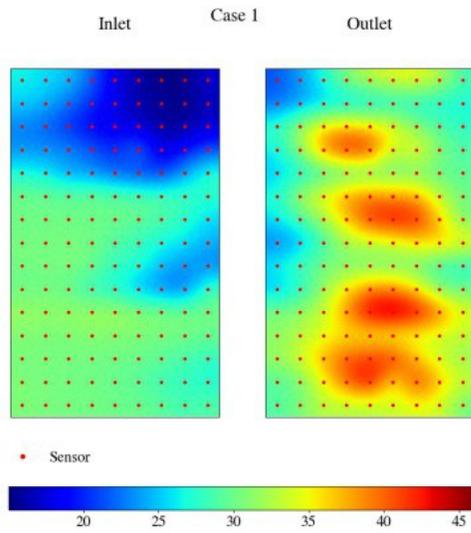


Figura 6.3: Medición de temperatura para el caso 1.

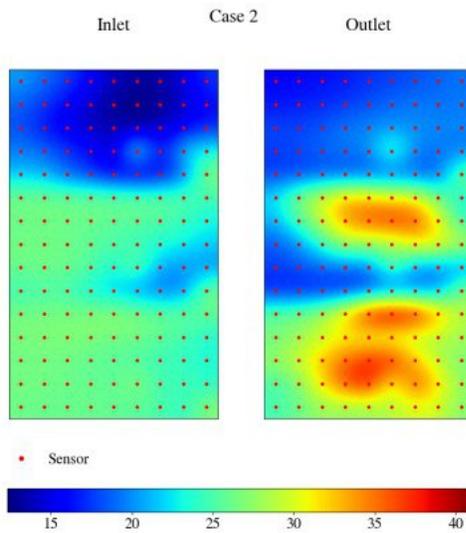


Figura 6.4: Medición de temperatura para el caso 2.

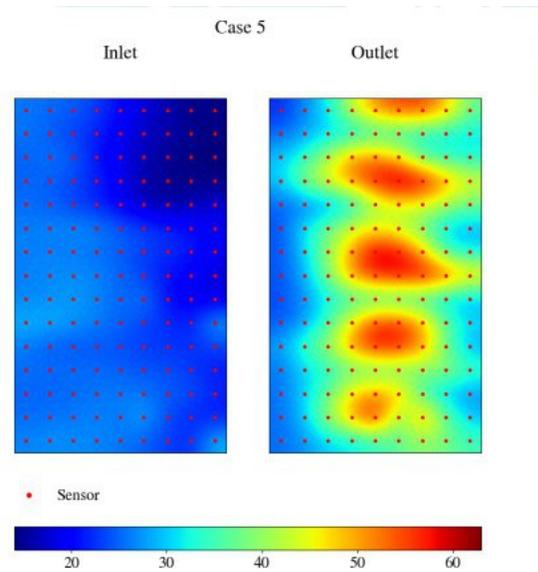


Figura 6.7: Medición de temperatura para el caso 5.

6.1.1. Visualización de datos

Se realizaron gráficas de la velocidad de ventilador de entrada y salida. Además, se realizaron gráficas del promedio de temperatura de las 135 mediciones de temperatura de entrada y salida a través del tiempo para cada caso. El caso 1 se utilizó en su totalidad para realizar las pruebas de los modelos, mientras que los casos 2 al 5 se dividieron en conjuntos de datos de entrenamiento y validación para utilizar durante el entrenamiento. En estos casos se utilizó el 80% de los datos para entrenamiento y 20% para validación. A continuación se muestran las figuras. Cabe resaltar que para las mediciones de velocidad de ventilador las gráficas de entrada y de salida son idénticas puesto que los valores eran iguales. En los casos de las mediciones de temperatura se puede observar que las mediciones temperatura son mayores a las de entrada. El caso 4 muestra la relación que existe entre las temperaturas y las velocidades de ventilador, puesto que al momento de incrementar el valor de PWM de la velocidad de ventilador se puede ver el impacto que tiene sobre la temperatura promedio de los sensores de salida.

Caso 1

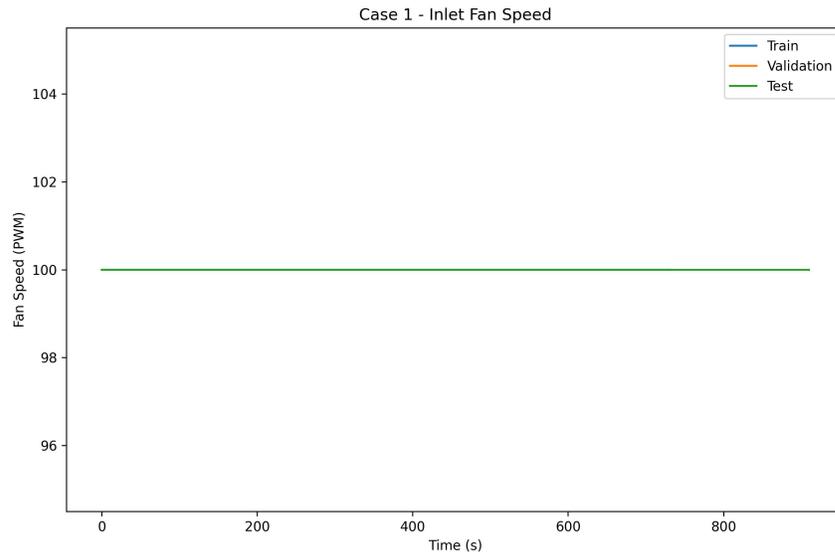


Figura 6.8: Velocidad de ventilador de entrada para el caso 1.

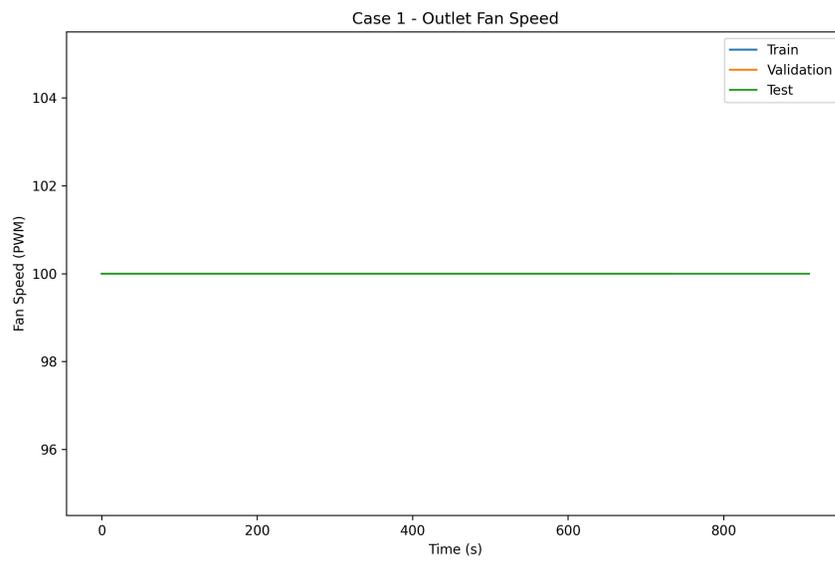


Figura 6.9: Velocidad de ventilador de salida para el caso 1.

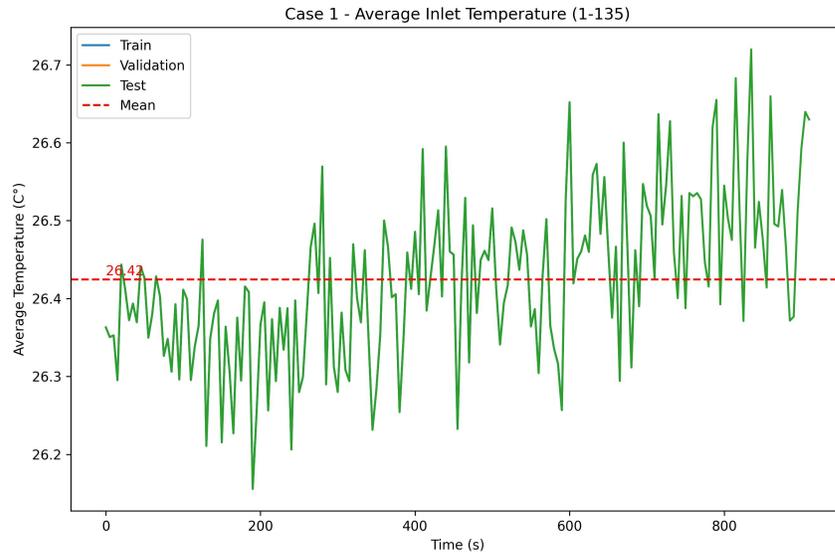


Figura 6.10: Media de temperatura de entrada para el caso 1.

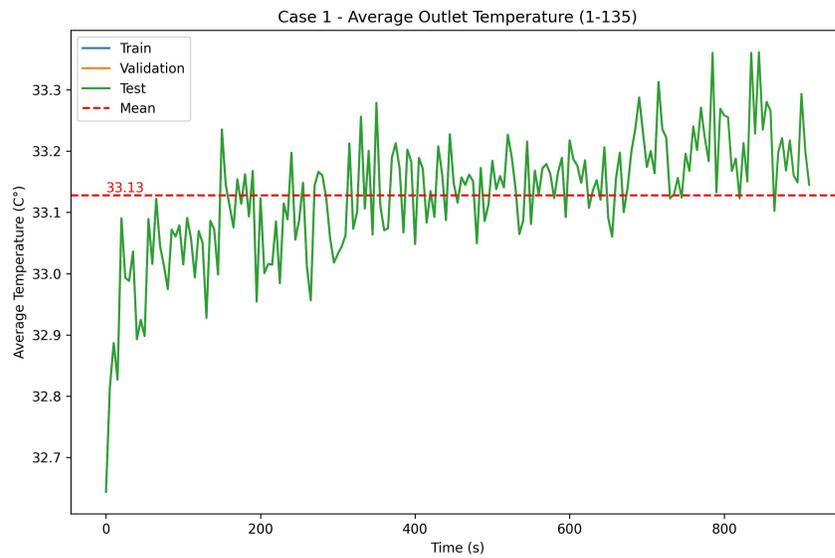


Figura 6.11: Media de temperatura de salida para el caso 1.

Caso 2

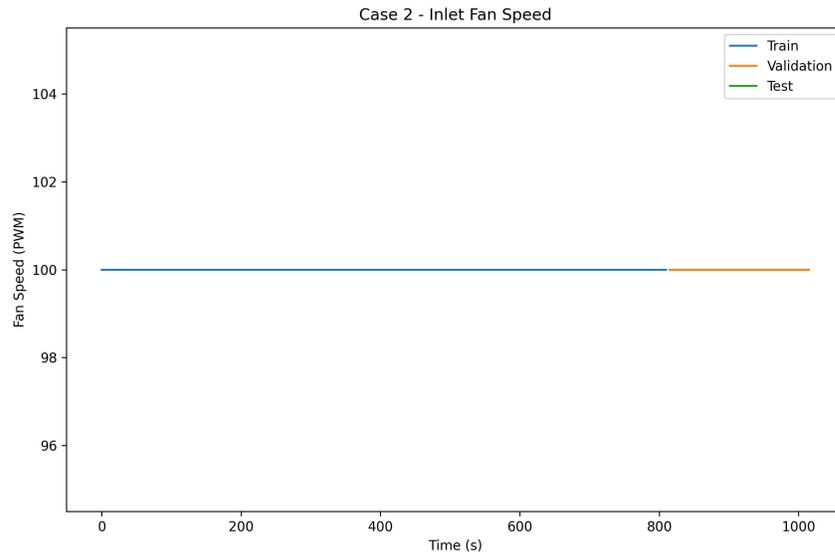


Figura 6.12: Velocidad de ventilador de entrada para el caso 2.

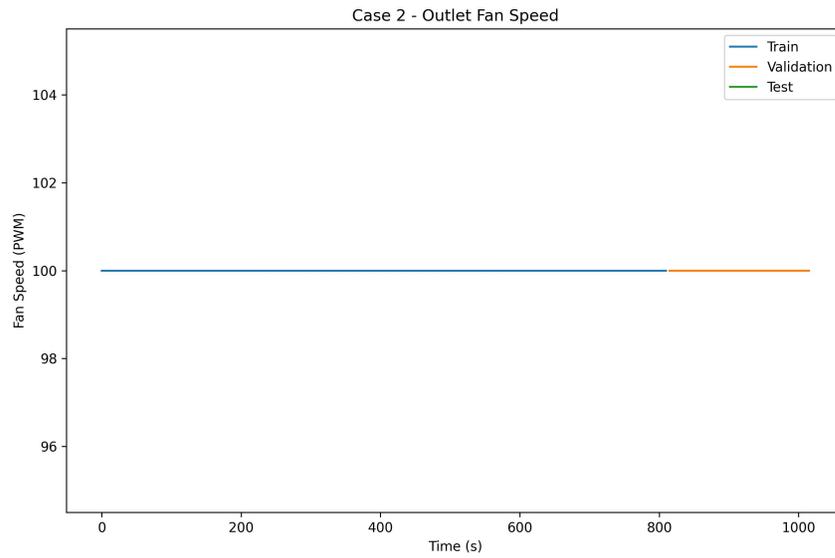


Figura 6.13: Velocidad de ventilador de salida para el caso 2.

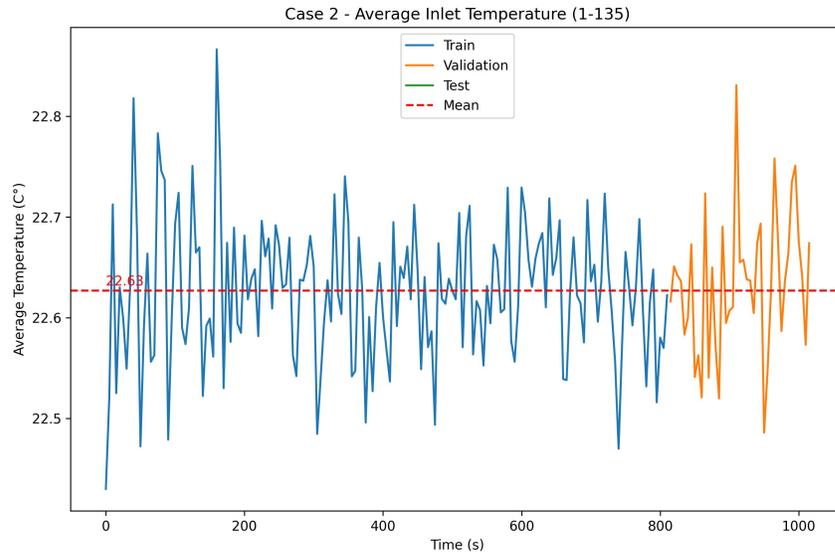


Figura 6.14: Media de temperatura de entrada para el caso 2.

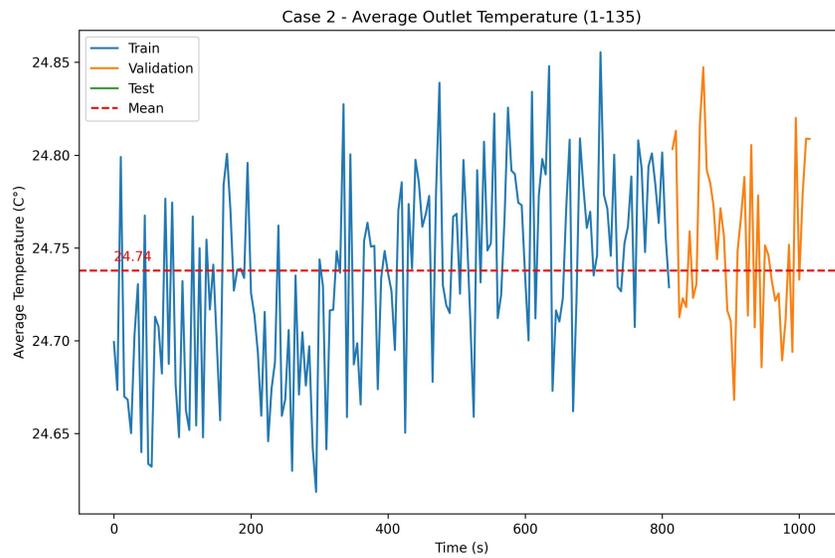


Figura 6.15: Media de temperatura de salida para el caso 2.

Caso 3

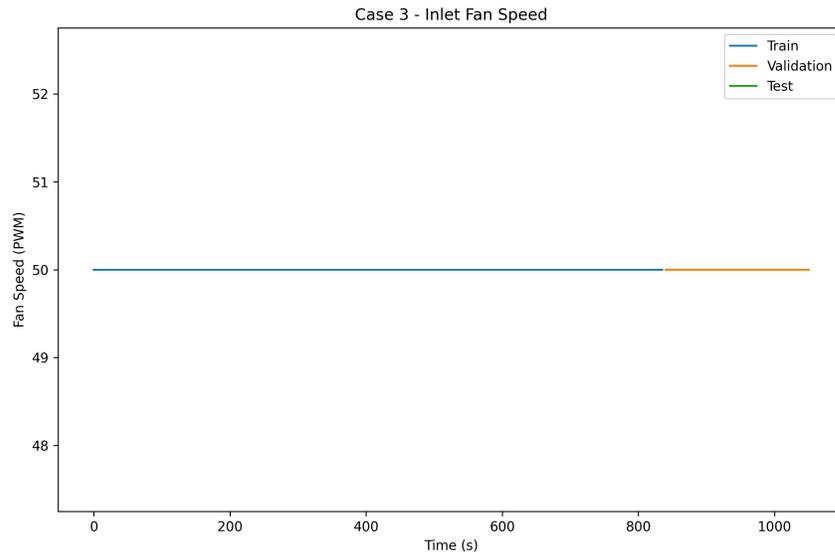


Figura 6.16: Velocidad de ventilador de entrada para el caso 3.

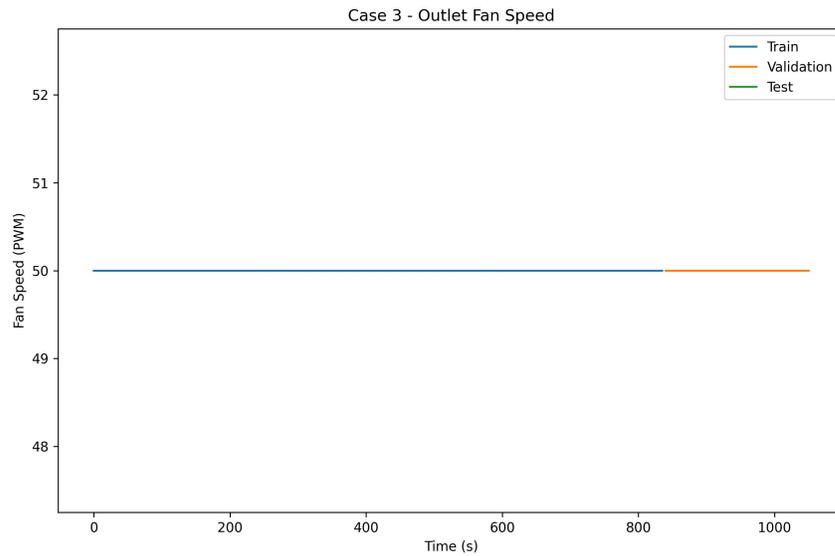


Figura 6.17: Velocidad de ventilador de salida para el caso 3.

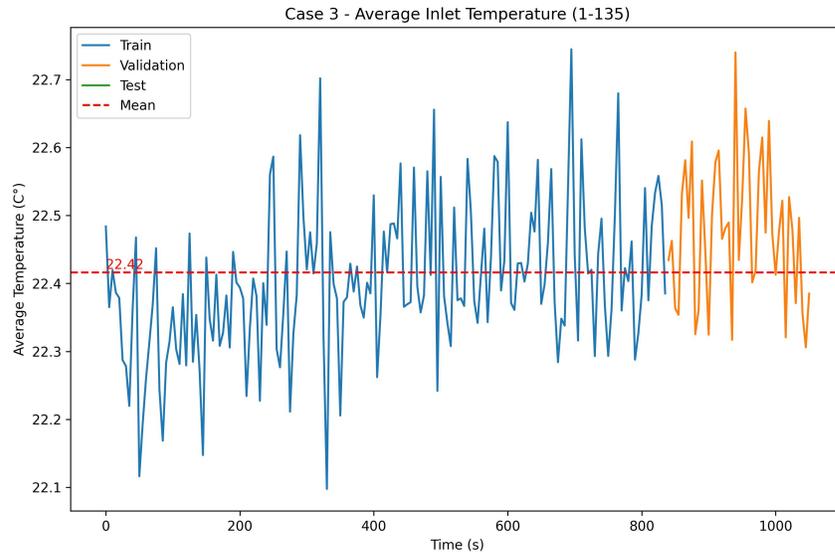


Figura 6.18: Media de temperatura de entrada para el caso 3.

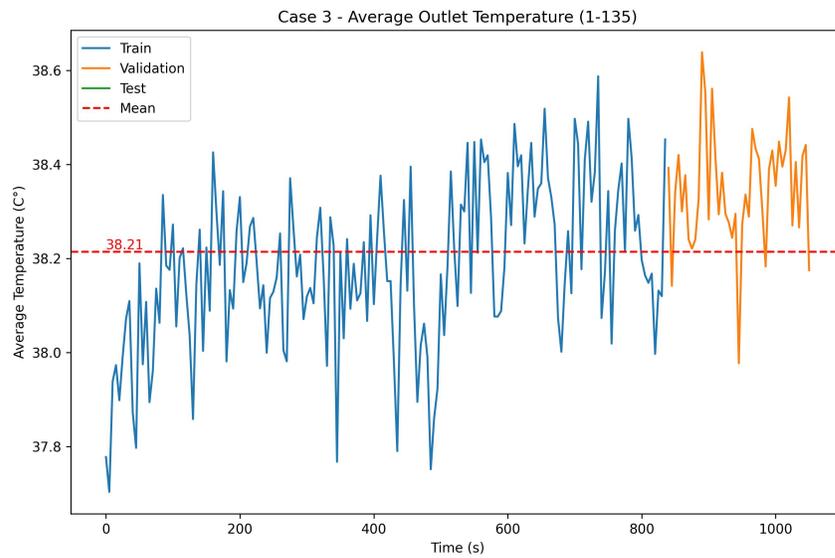


Figura 6.19: Media de temperatura de salida para el caso 3.

Caso 4

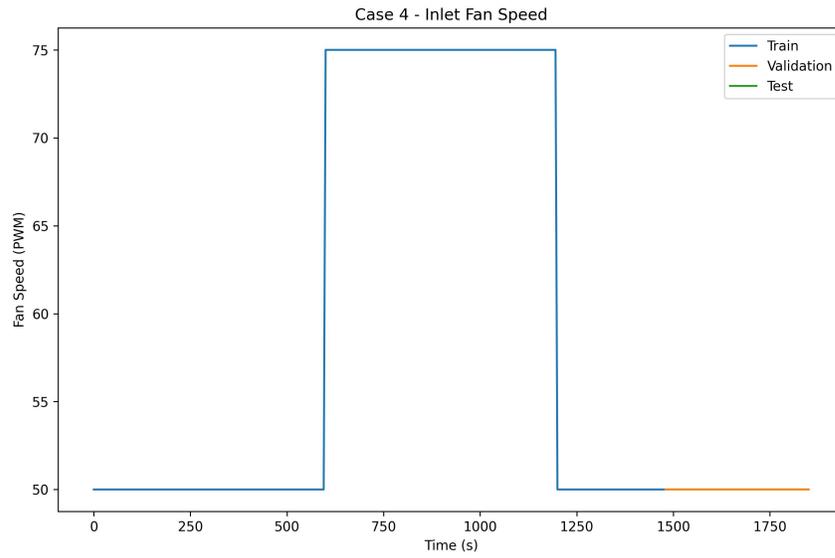


Figura 6.20: Velocidad de ventilador de entrada para el caso 4.

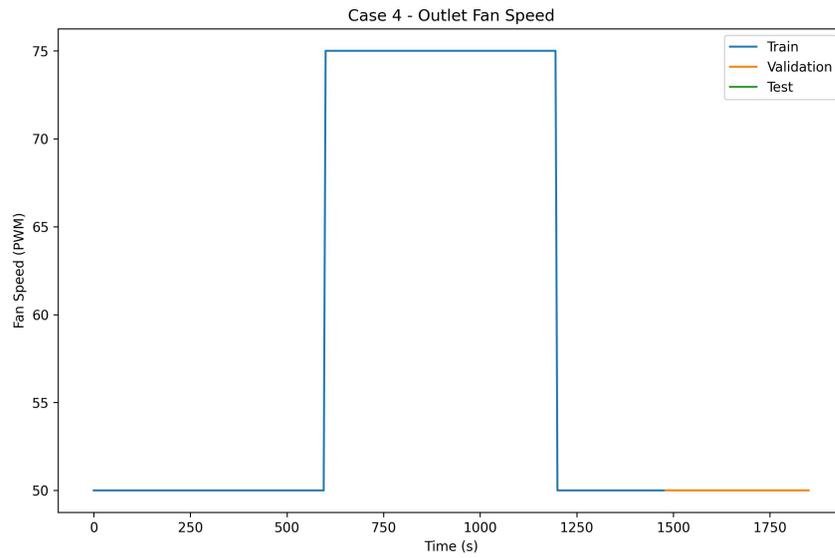


Figura 6.21: Velocidad de ventilador de salida para el caso 4.

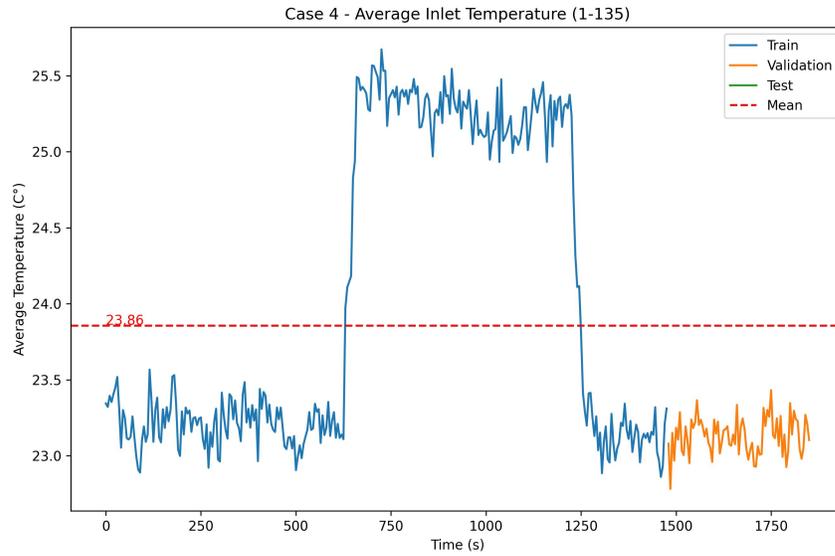


Figura 6.22: Media de temperatura de entrada para el caso 4.

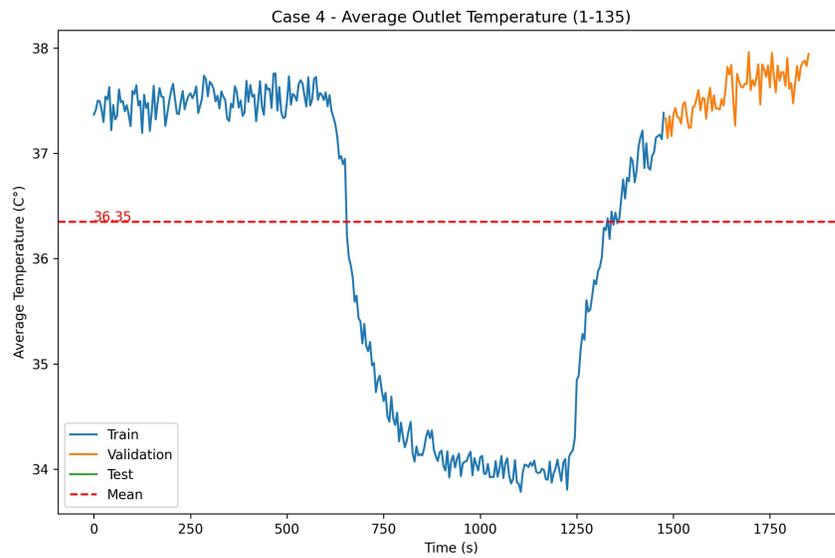


Figura 6.23: Media de temperatura de salida para el caso 4.

Caso 5

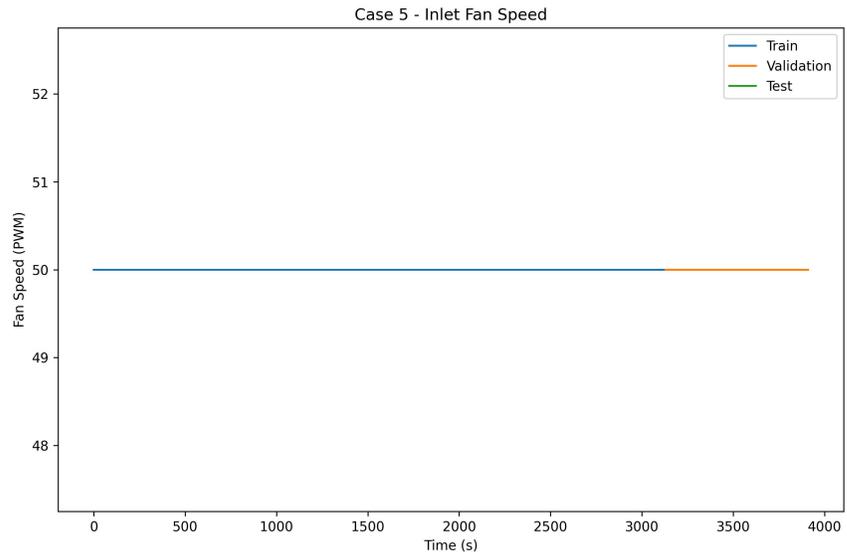


Figura 6.24: Velocidad de ventilador de entrada para el caso 5.

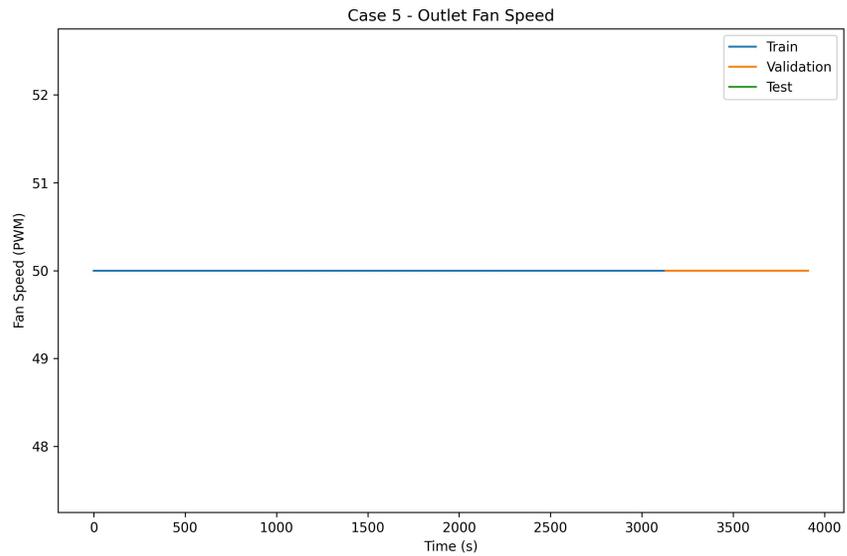


Figura 6.25: Velocidad de ventilador de salida para el caso 5.

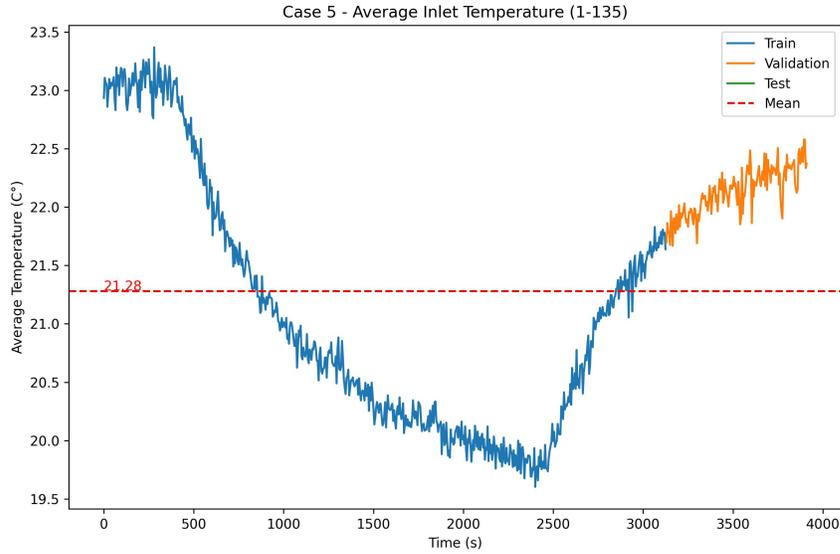


Figura 6.26: Media de temperatura de entrada para el caso 5.

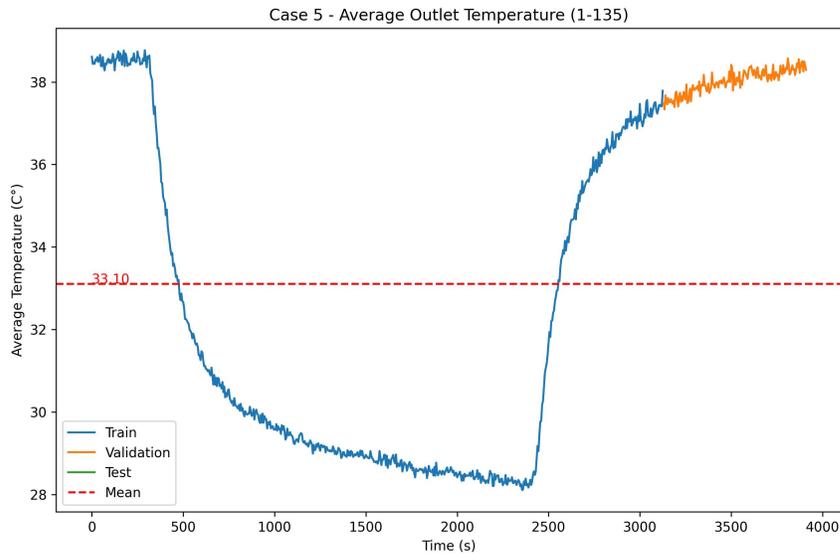


Figura 6.27: Media de temperatura de salida para el caso 5.

6.2. Preprocesamiento

Para las columnas de velocidad de ventilador de entrada y de salida se aplicó una normalización de escalamiento con valores mínimos y máximos. Los valores máximos y mínimos de los valores de PWM son de 100 y 0, debido a la naturaleza de la medición. Por lo tanto, la transformación de los datos se representa por la siguiente ecuación:

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} = \frac{x}{100}$$

Con esto, se logró que los valores se encuentren en un rango de 0 a 1.

De igual forma, para las variables de temperatura se aplicó una normalización de escalamiento con valores mínimos y máximos. En este caso, se utilizó los valores mínimos y máximos de todo el conjunto de datos (tomando en consideración todos los casos), que son 9.08 y 64.7 respectivamente. Por lo tanto la transformación de los datos se representa por la siguiente ecuación:

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} = \frac{x - 9.08}{64.7 - 9.08}$$

Además, se utilizó el filtro de Savitsky-Golay para alisar la data de las mediciones de temperatura. Esto con el objetivo de incrementar la precisión de los datos sin afectar la tendencia de la misma. Se utilizó el filtro con una ventana de 9 observaciones y con grado 3. A continuación se muestran las gráficas generadas luego de la aplicación del filtro y el preprocesamiento.

Caso 1

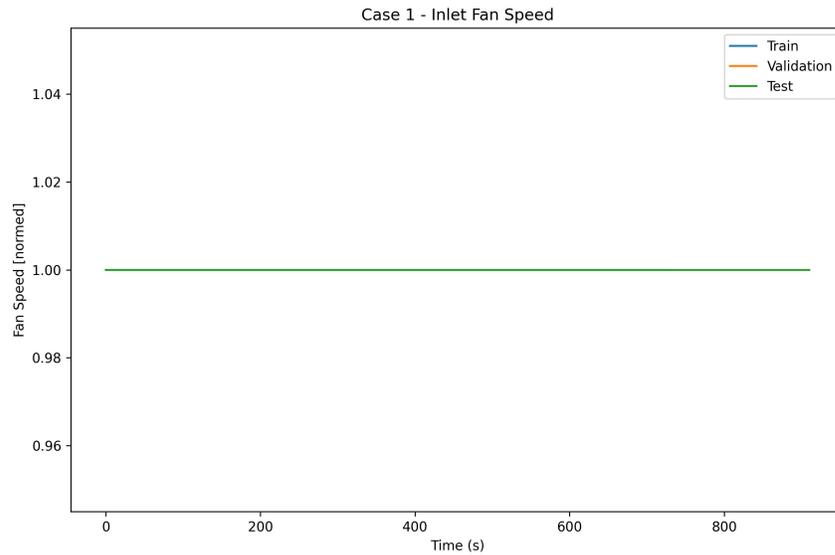


Figura 6.28: Velocidad de ventilador de entrada normalizada para el caso 1.

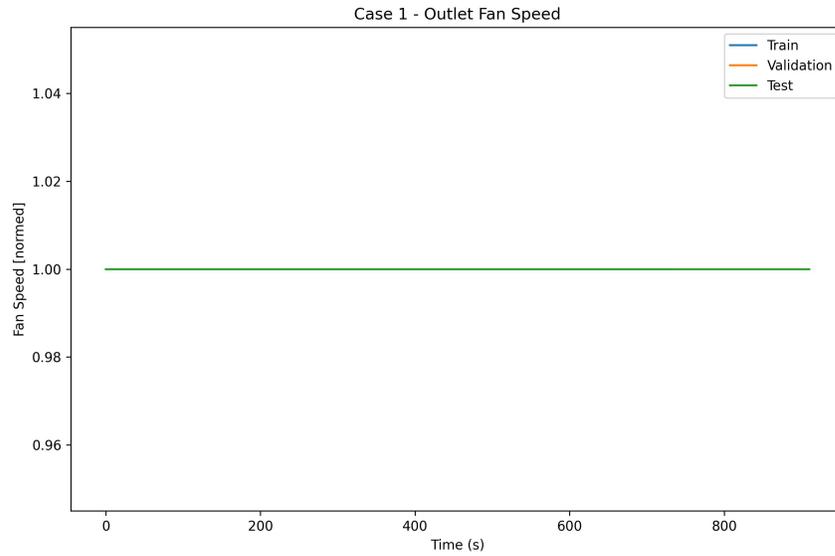


Figura 6.29: Velocidad de ventilador de salida normalizada para el caso 1.

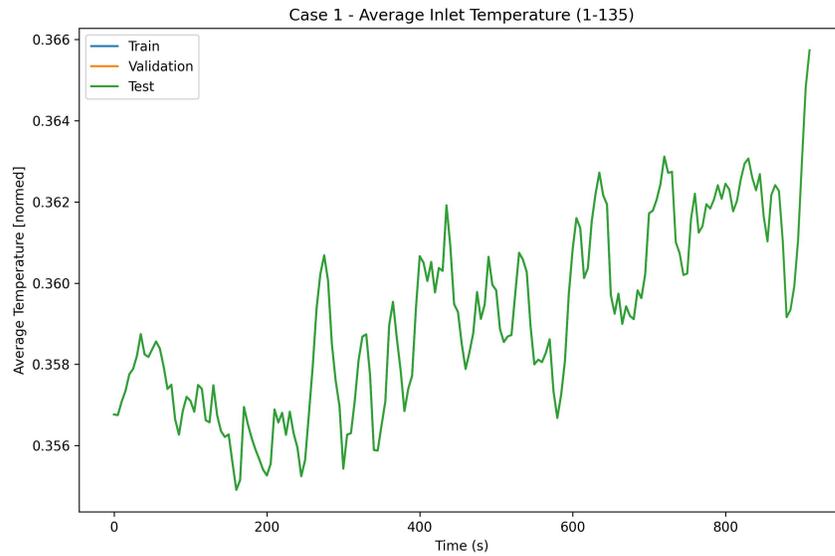


Figura 6.30: Media de temperatura de entrada normalizada para el caso 1.

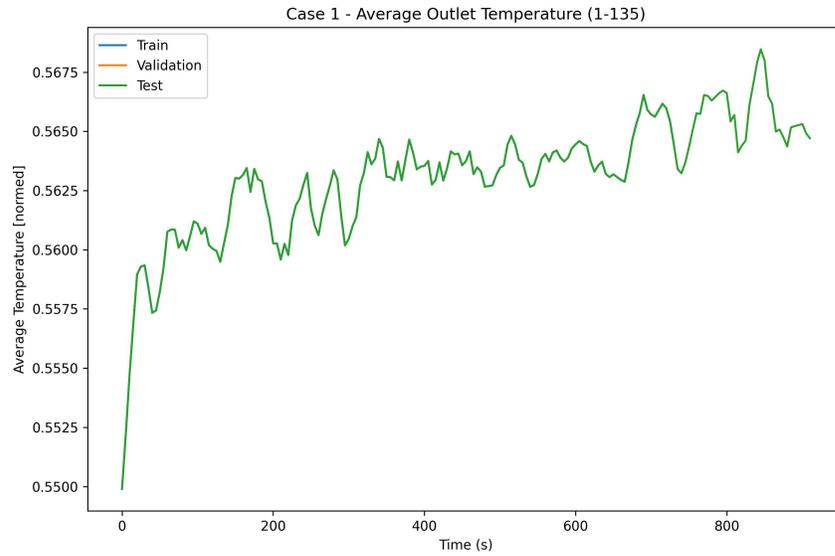


Figura 6.31: Media de temperatura de salida normalizada para el caso 1.

Caso 2

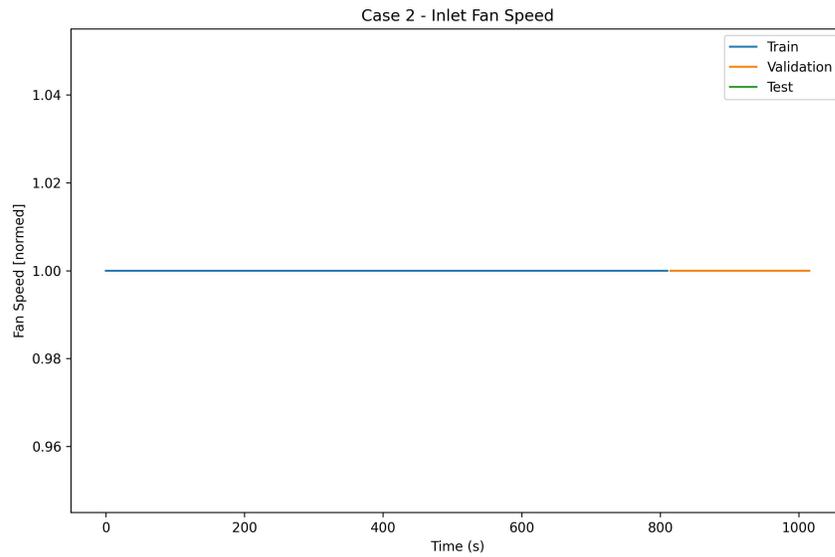


Figura 6.32: Velocidad de ventilador de entrada normalizada para el caso 2.

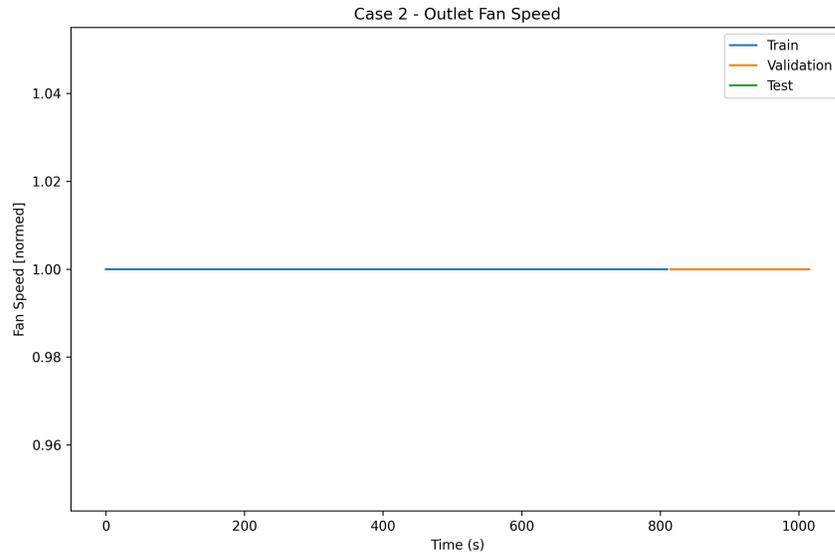


Figura 6.33: Velocidad de ventilador de salida normalizada para el caso 2.

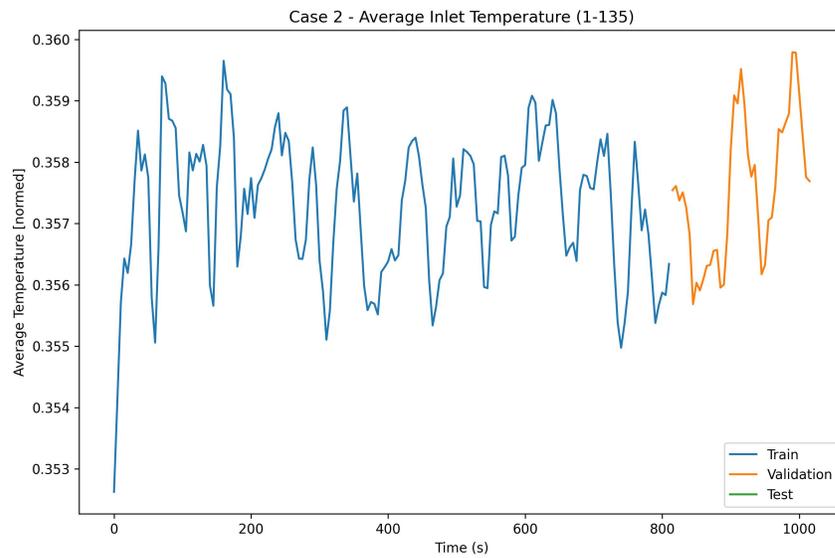


Figura 6.34: Media de temperatura de entrada normalizada para el caso 2.

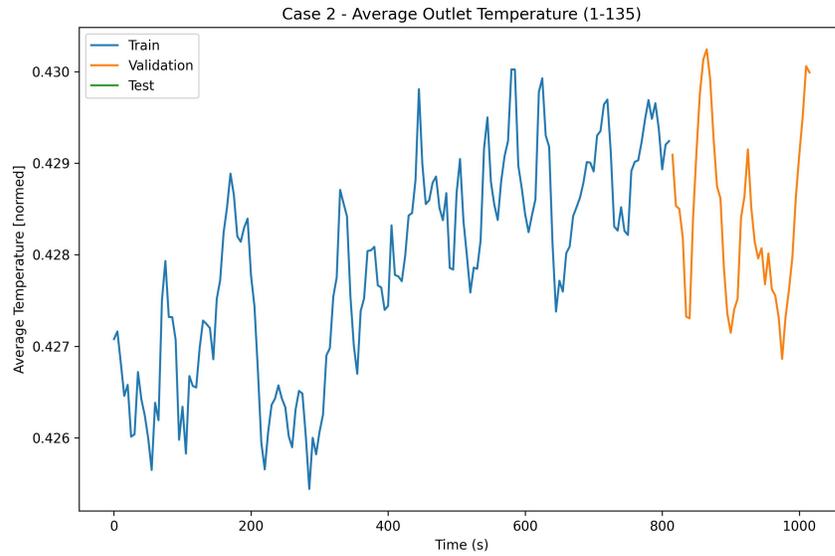


Figura 6.35: Media de temperatura de salida normalizada para el caso 2.

Caso 3

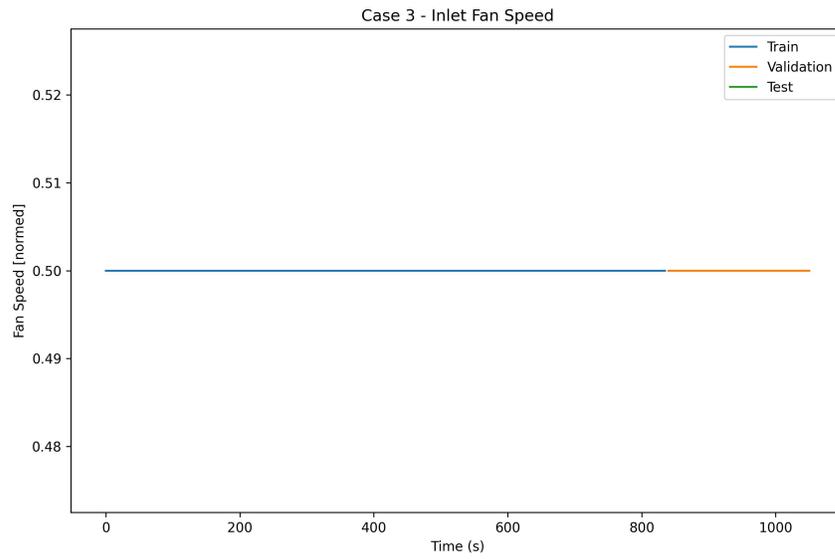


Figura 6.36: Velocidad de ventilador de entrada normalizada para el caso 3.

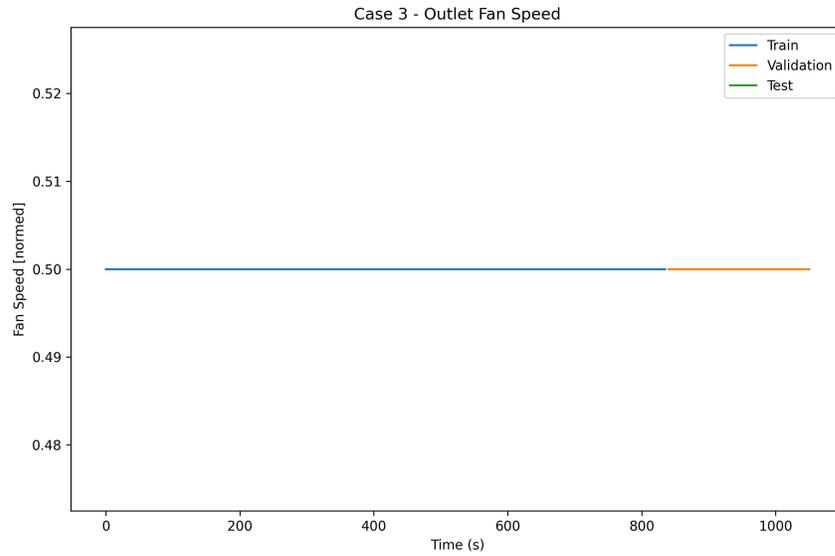


Figura 6.37: Velocidad de ventilador de salida normalizada para el caso 3.

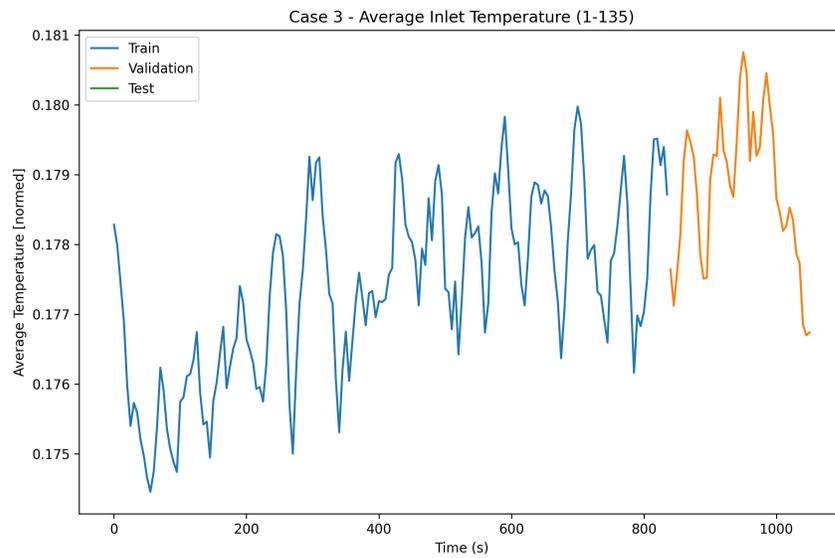


Figura 6.38: Media de temperatura de entrada normalizada para el caso 3.

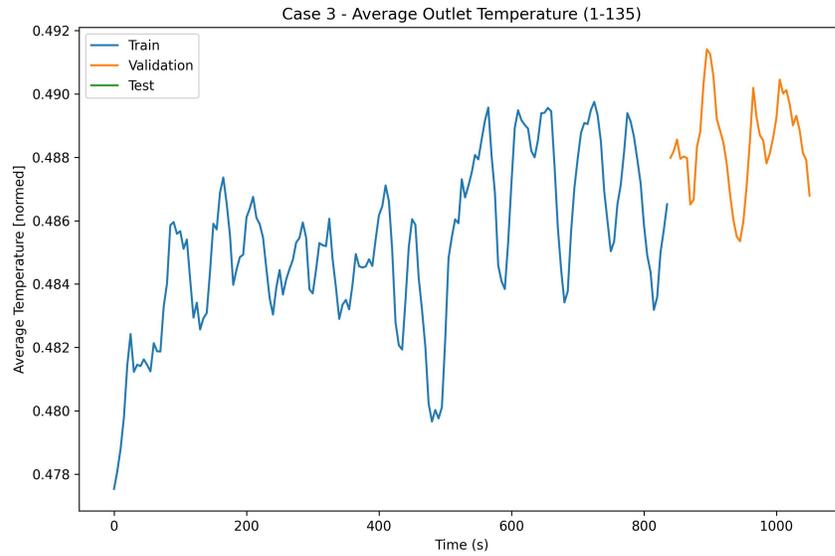


Figura 6.39: Media de temperatura de salida normalizada para el caso 3.

Caso 4

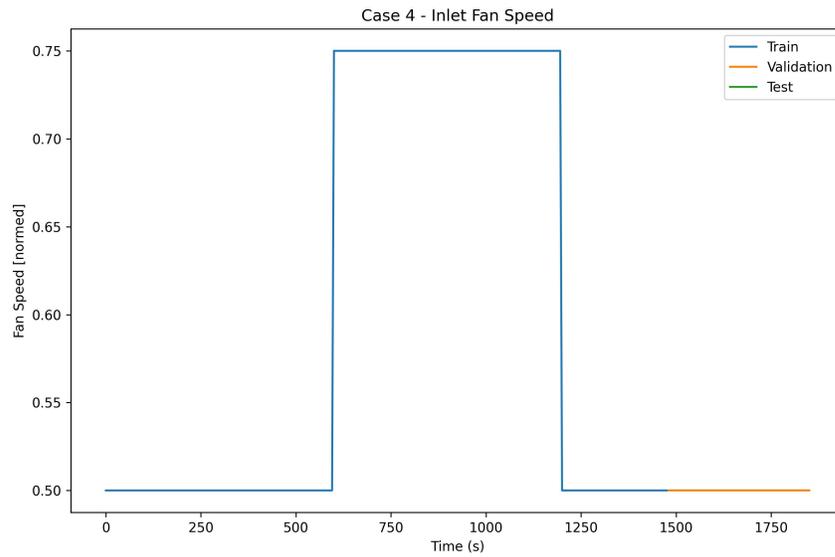


Figura 6.40: Velocidad de ventilador de entrada normalizada para el caso 4.

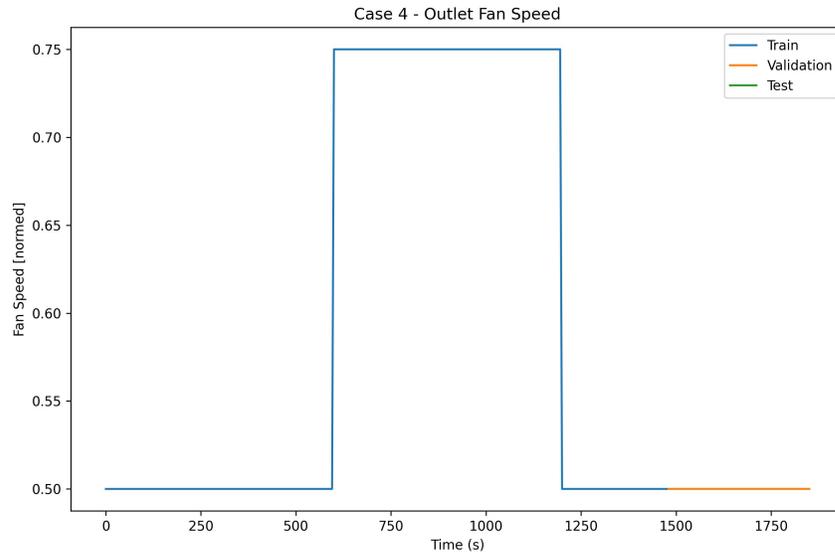


Figura 6.41: Velocidad de ventilador de salida normalizada para el caso 4.

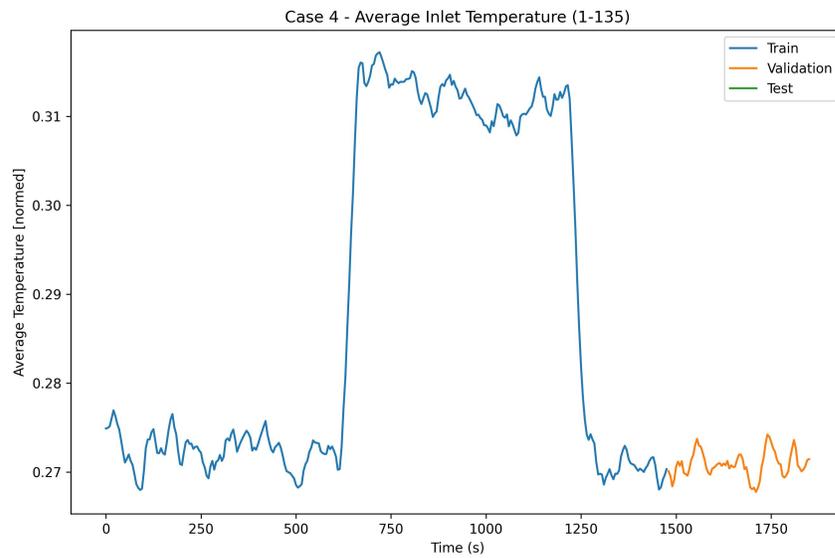


Figura 6.42: Media de temperatura de entrada normalizada para el caso 4.

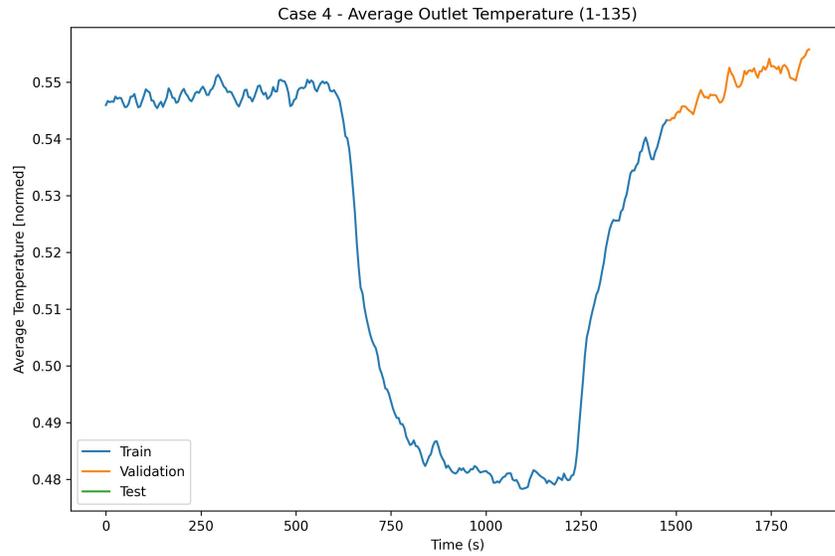


Figura 6.43: Media de temperatura de salida normalizada para el caso 4.

Caso 5

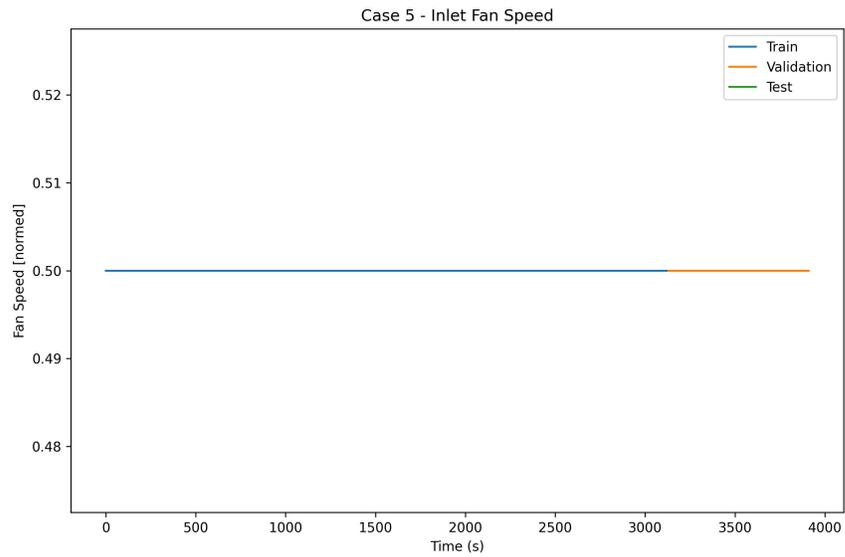


Figura 6.44: Velocidad de ventilador de entrada normalizada para el caso 5.

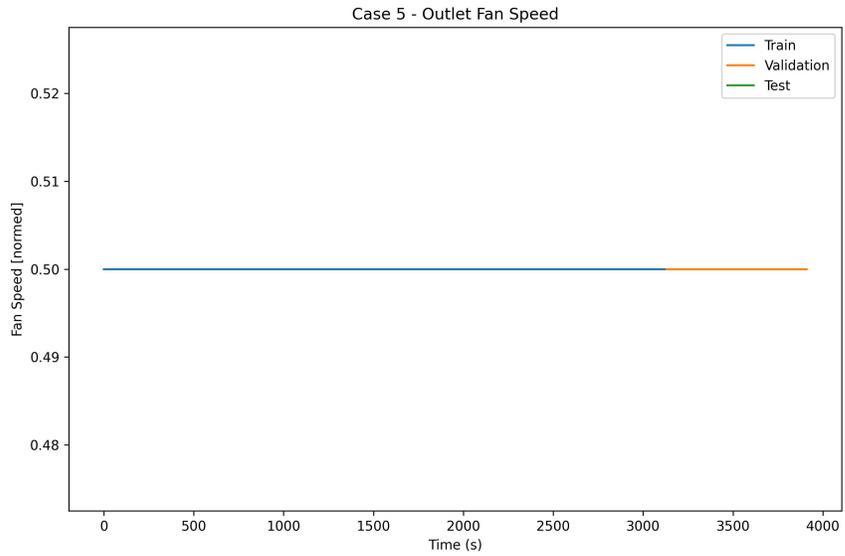


Figura 6.45: Velocidad de ventilador de salida normalizada para el caso 5.

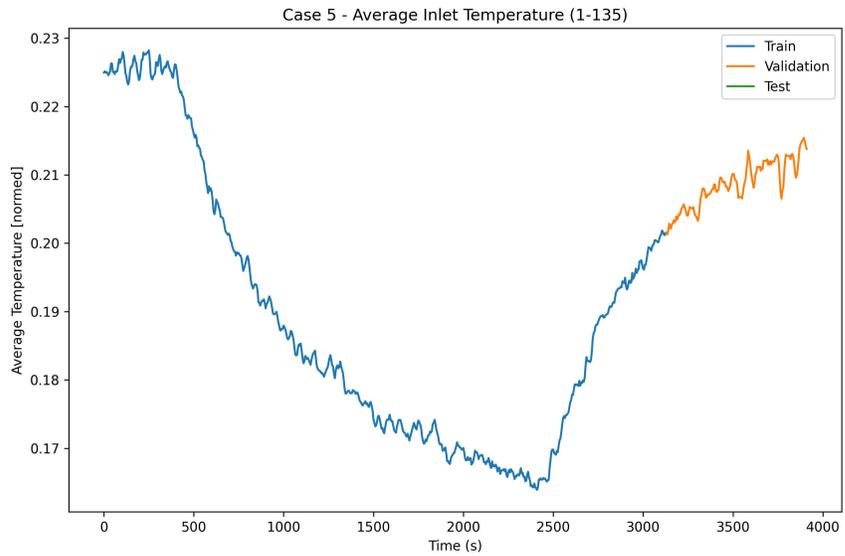


Figura 6.46: Media de temperatura de entrada normalizada para el caso 5.

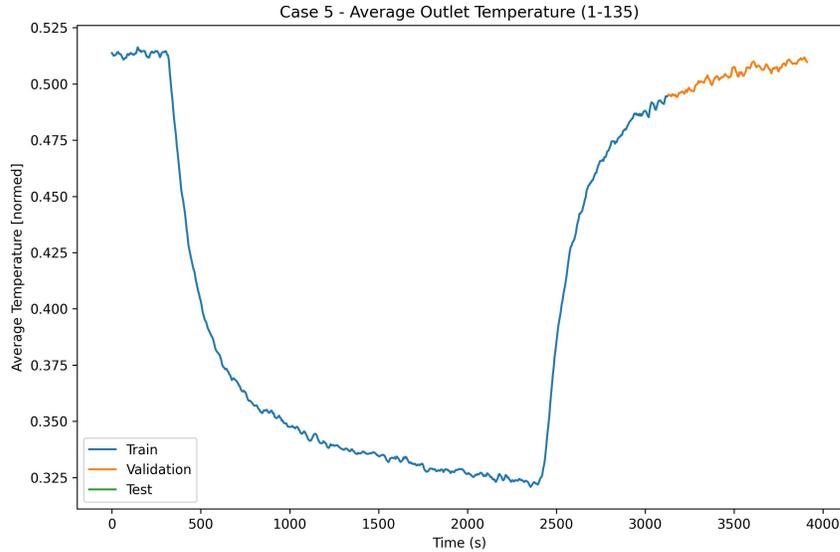


Figura 6.47: Media de temperatura de salida normalizada para el caso 5.

Como último paso del preprocesamiento, se realizó el *windowing* para cada caso del conjunto de datos. Este crea ventanas que utilizan 18 observaciones para predecir 18 observaciones en el futuro. Esto significa que se utilizan 1 min 30 s de datos de entrada para predecir 1 min 30 s en el futuro. Para la mayoría de modelos, exceptuando el modelo autorregresivo, se utilizaron las variables de velocidad de ventilador de entrada y salida, las 135 mediciones de temperatura de entrada y las 135 mediciones de temperatura de salida como variables de entrada (272 entradas) y las 135 mediciones de temperatura de salida como variables de salida. Esto se puede interpretar como utilizar toda la información del pasado disponible (velocidad de ventilador y temperatura) para poder predecir solamente la temperatura de salida en el futuro. En el caso del modelo autorregresivo se utilizaron todas las variables de temperatura (entrada y salida), tanto de entrada como salida del modelo, debido a la naturaleza de este tipo de modelos. También se utilizó un valor de batch de 32 al momento de generar el conjunto de datos con el *windowing* aplicado. Este valor se seleccionó después de probar diferentes valores y comparar los resultados obtenidos y la velocidad de entrenamiento. A continuación, en la Figura 6.48, se muestra un ejemplo de ventana para la variable de salida de temperatura en la posición 60. De igual forma, se crean ventanas de esta forma para todas las variables contenidas en el conjunto de datos de tal forma que se tengan todas las variables preprocesadas para alimentar al modelo.

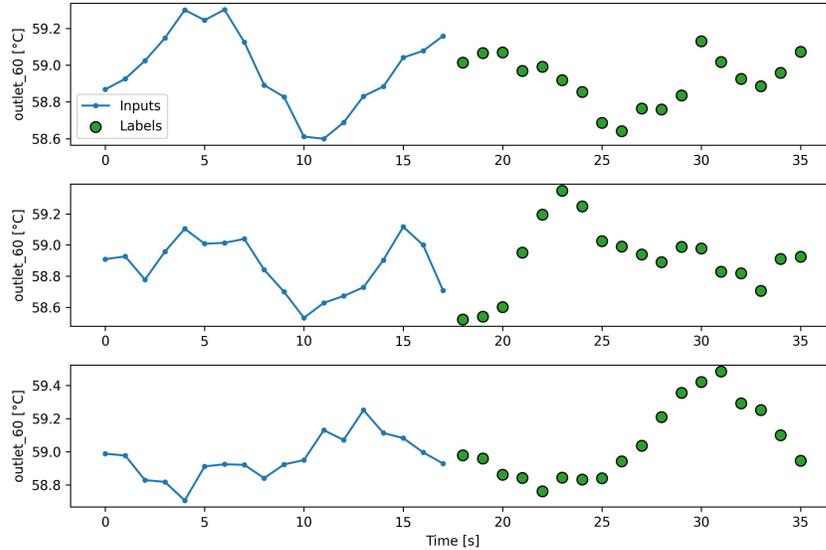


Figura 6.48: Ventana de temperatura de salida en posición 60.

6.3. Modelo lineal base

Como modelo base se diseñó e implementó un modelo lineal. Este consiste únicamente de la capa de entrada, una capa de filtro, la capa densa de salida y una capa para darle la forma correcta a la salida. Cabe destacar que a pesar de tener 18 pasos en el tiempo como entrada, solamente se propaga el último. Esto debido a que este tipo de modelos no es capaz de identificar la adyacencia temporal de estas observaciones. Sumado a esto, se utilizó el optimizador Adam y el error cuadrático medio como función de pérdida y se entrenó durante 15 épocas. Como se mencionó anteriormente, se utilizaron los casos del 2 al 5 para el entrenamiento y el caso 1 para probar el modelo. La siguiente Figura (6.49) muestra los detalles de la arquitectura.

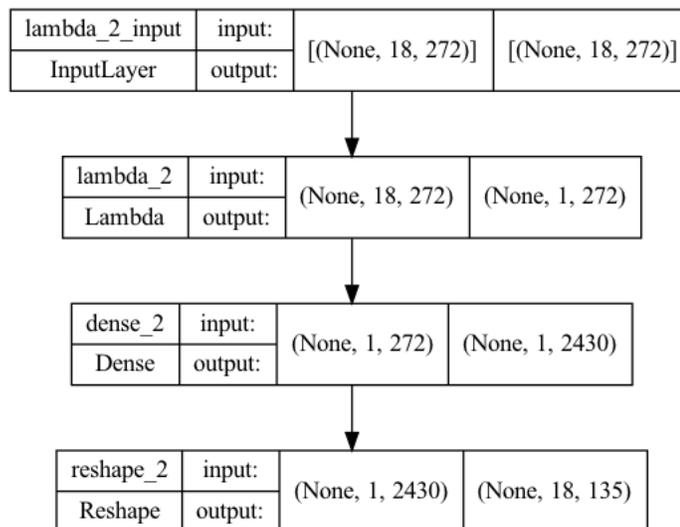


Figura 6.49: Modelo lineal base

6.3.1. Resultados

Este modelo tuvo un tiempo de entrenamiento total de **24.81 s**. A continuación, en la Tabla 6.2 se muestra el tiempo de entrenamiento por caso con la red lineal.

Caso	Tiempo
Caso 2	6.08 s
Caso 3	5.69 s
Caso 4	6.03 s
Caso 5	7.01 s

Tabla 6.2: Tiempo de entrenamiento por caso para el modelo lineal.

En la Tabla 6.3 se muestra los resultados obtenidos en los diferentes casos y también con el conjunto de datos de prueba. De estos, vale la pena destacar los valores con el conjunto de datos de prueba: pérdida de **10.01** y RMSE de **3.20 °C**. Finalmente, en la Figura 6.50 se muestra un ejemplo de la forma en la que el modelo que realiza las predicciones. Cabe resaltar que esta es solamente una de las variables (posición 60 de temperatura de salida) de todas las que tiene el modelo como salida. En este caso, el modelo hace predicciones de la misma forma para las otras 135 posiciones de la temperatura de salida y con base a todas estas predicciones es que se calcula la pérdida. Se muestra solamente una variable en este caso para facilitar la comprensión de la visualización. En los modelos posteriores se realizan gráficas de predicciones de la misma manera y se puede asumir que solamente es una de las variables de la salida multivariada que producen los modelos.

Caso	Pérdida	RMSE	Pérdida (validación)	RMSE (validación)
Caso 2	9.09	3.02	9.09	3.02
Caso 3	9.09	3.02	9.09	3.01
Caso 4	9.11	3.02	9.10	3.02
Caso 5	9.11	3.02	9.09	3.02
Prueba	10.01	3.20	N/A	N/A

Tabla 6.3: Resultados de modelo lineal.

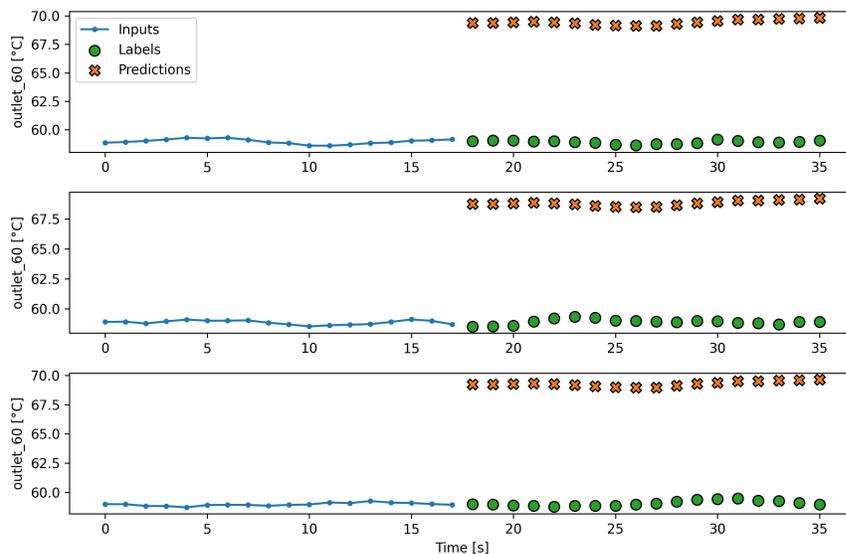


Figura 6.50: Predicciones de modelo lineal.

6.4. Modelo con capas densas

Se realizó un modelo con capas densas, que está conformado por dos capas densas con regularizador L2 con un valor de 0.015 y función de activación ReLU. Este modelo también necesita de una capa de filtro (para propagar solamente la última observación a las siguientes capas) y la capa final para darle la forma correcta a la salida. Sumado a esto, contiene capas de *dropout* con un valor de 0.3, que junto al regularizador L2 tienen propósito de evitar el sobreajuste del modelo. Se entrenó durante 30 épocas. Como en el modelo lineal, también se utilizó el optimizador Adam y el error cuadrático medio como función de pérdida. En la Figura [6.51](#) se muestran los detalles de la arquitectura. Como se mencionó anteriormente, se utilizaron los casos del 2 al 5 para el entrenamiento y el caso 1 para prueba y evaluación del modelo.

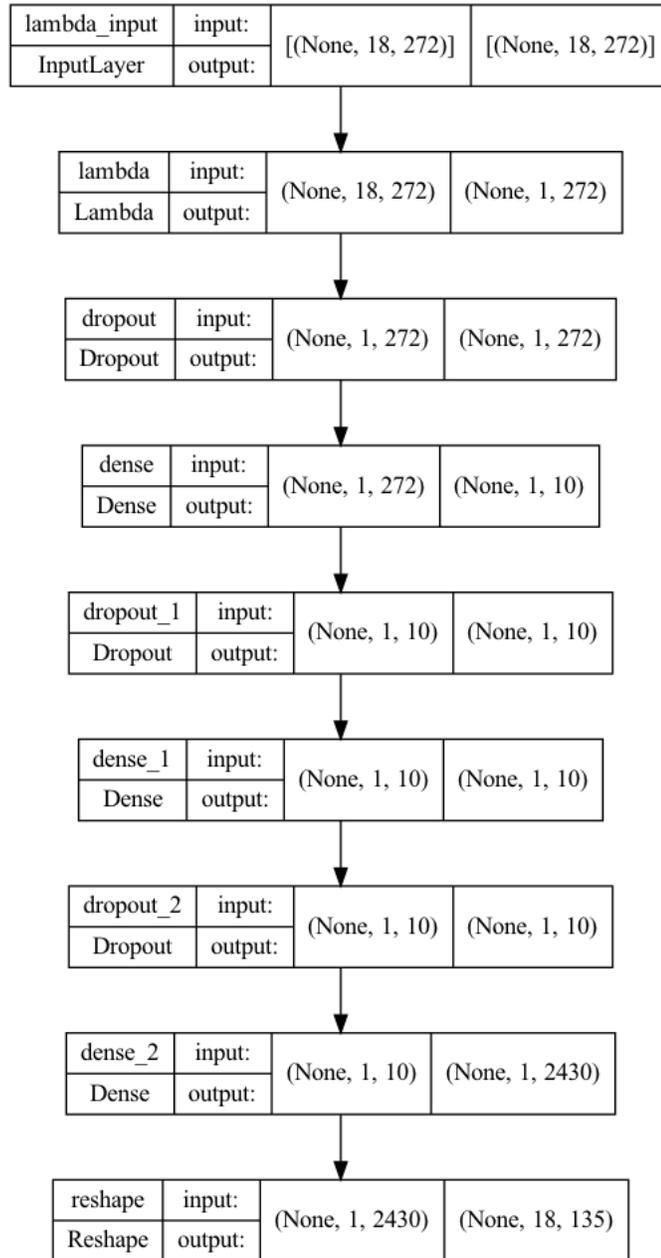


Figura 6.51: Modelo ANN

6.4.1. Resultados

El modelo tuvo un tiempo de entrenamiento total de **55.28 s**. A continuación, en la Tabla [6.4](#) se muestra el tiempo de entrenamiento por caso con la red lineal.

De igual forma, en la Tabla [6.5](#) se muestra los resultados obtenidos en los diferentes casos y también con el conjunto de datos de prueba. De estos, vale la pena destacar los valores con el conjunto de datos de prueba: pérdida de **11.21** y RMSE de **3.39 °C**. Finalmente, en la Figura [6.52](#) se muestra un ejemplo de las predicciones que realiza el modelo (con la variable de temperatura de

Caso	Tiempo
Caso 2	12.57 s
Caso 3	11.74 s
Caso 4	13.33 s
Caso 5	17.64 s

Tabla 6.4: Tiempo de entrenamiento por caso para el modelo con capas densas.

salida en la posición 60).

Caso	Pérdida	RMSE	Pérdida (validación)	RMSE (validación)
Caso 2	15.33	3.10	15.57	3.18
Caso 3	11.31	3.06	11.77	3.16
Caso 4	9.74	3.05	10.33	3.17
Caso 5	9.56	3.07	10.18	3.19
Prueba	11.21	3.39	N/A	N/A

Tabla 6.5: Resultados de modelo con capas densas.

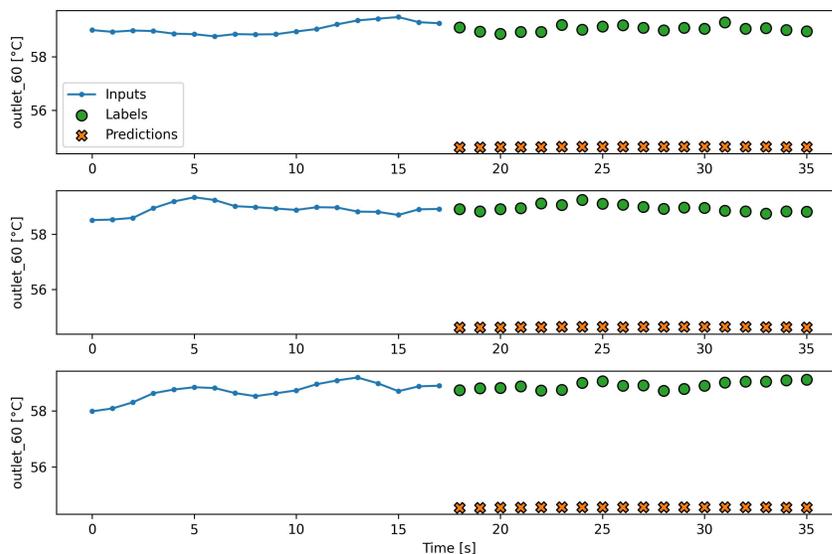


Figura 6.52: Predicciones de modelo con capas densas.

6.5. Modelo de red neuronal convolucional

De igual forma que los modelos anteriores, este modelo utiliza ventanas de 18 observaciones de entrada para predecir 18 observaciones en el futuro. Esta red está compuesta por una sola capa de convolución de una dimensión y una capa densa de salida. La principal diferencia entre este modelo y los anteriores es que este modelo permite propagar más observaciones a las siguientes capas. En este caso se propagan las últimas 3 observaciones. Este modelo contiene capas de *dropout* con un valor de 0.2 y la capa de convolución con valor de filtros de 256, tamaño de kernel de 3 y función de activación ReLU y se entrenó durante 15 épocas. De igual forma que los modelos anteriores, se utilizó el optimizador Adam y el error cuadrático medio como función de pérdida. A continuación

(Figura 6.53) se muestran los detalles de la misma.

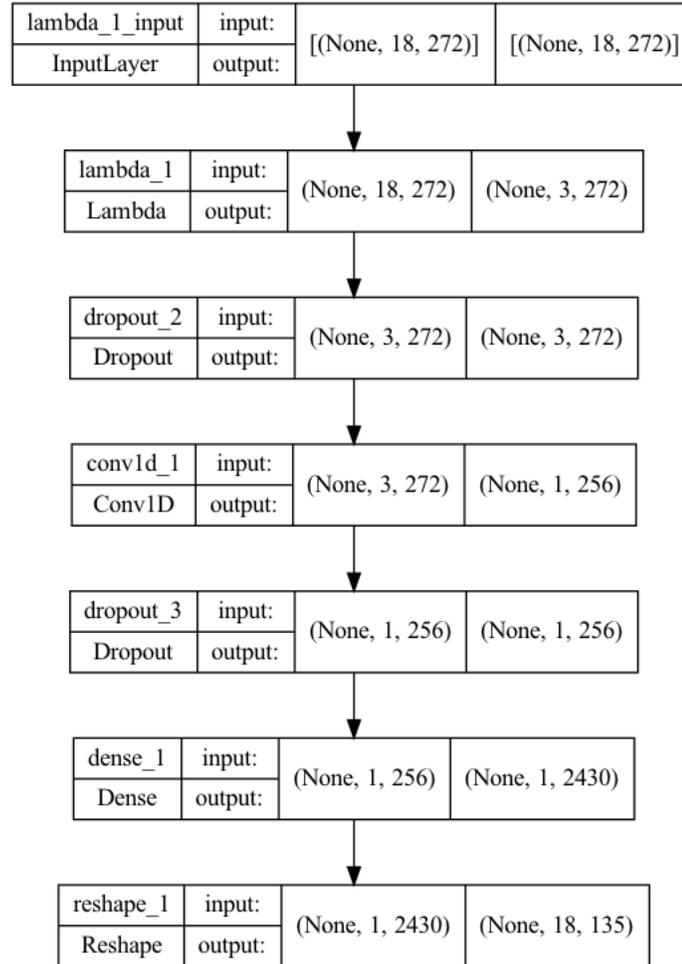


Figura 6.53: Modelo CNN

6.5.1. Resultados

Este modelo tuvo un tiempo de entrenamiento total de **27.08 s**. A continuación, en la Tabla 6.6 se muestra el tiempo de entrenamiento por caso con la red lineal.

Caso	Tiempo
Caso 2	6.61 s
Caso 3	6.04 s
Caso 4	6.72 s
Caso 5	7.71 s

Tabla 6.6: Tiempo de entrenamiento por caso para el modelo convolucional.

De igual forma, en la Tabla 6.7 se muestra los resultados obtenidos en los diferentes casos y también con el conjunto de datos de prueba. De estos, vale la pena destacar los valores con el conjunto de datos de prueba: pérdida de **10.61** y RMSE de **3.31 °C**. Finalmente, en la Figura 6.54

se muestra un ejemplo de las predicciones que realiza el modelo (con la variable de temperatura de salida en la posición 60).

Caso	Pérdida	RMSE	Pérdida (validación)	RMSE (validación)
Caso 2	9.09	3.01	9.09	3.02
Caso 3	9.09	3.02	9.09	3.01
Caso 4	9.13	3.02	9.11	3.02
Caso 5	9.11	3.02	9.10	3.02
Prueba	10.61	3.31	N/A	N/A

Tabla 6.7: Resultados de modelo convolucional.

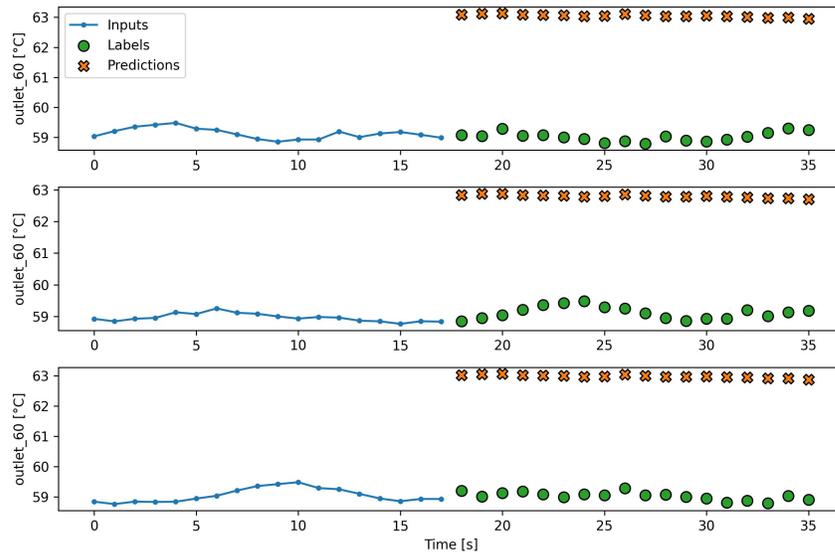


Figura 6.54: Predicciones de modelo convolucional.

6.6. Modelo de red neuronal recurrente

Se creó una red LSTM *single-shot* que predice 18 observaciones en el futuro con 18 observaciones de entrada. Esta consiste de una capa LSTM con 32 unidades, una capa de *dropout* (con valor de 0.5) y la salida. Se utilizó el optimizador Adam y el error cuadrático medio como función de pérdida. Esta se entrenó con 30 épocas. A continuación, en la Figura [6.55](#) se muestran los detalles de la arquitectura.

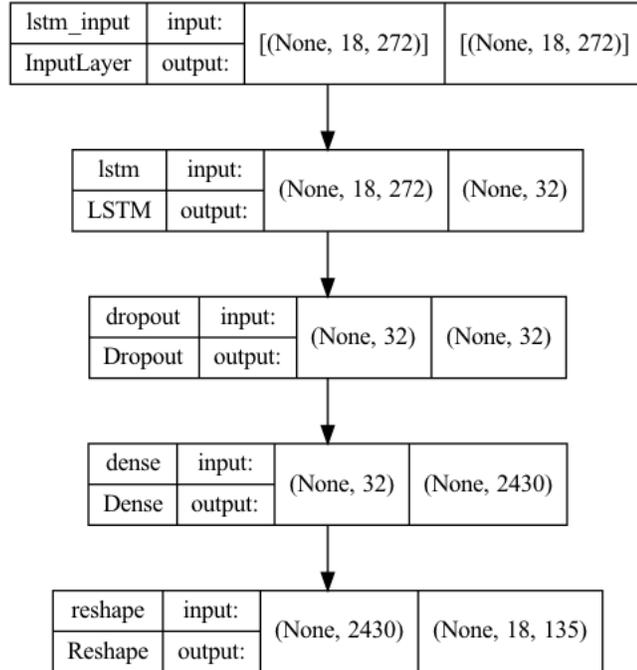


Figura 6.55: RNN con capa LSTM.

6.6.1. Resultados

Este modelo tuvo un tiempo de entrenamiento total de **56.57 s**. A continuación, en la Tabla 6.9 se muestra el tiempo de entrenamiento por caso con la red recurrente.

Caso	Tiempo
Caso 2	13.81 s
Caso 3	12.02 s
Caso 4	13.13 s
Caso 5	17.61 s

Tabla 6.8: Tiempo de entrenamiento por caso para el modelo RNN.

De igual forma, en la Tabla 6.9 se muestra los resultados obtenidos en los diferentes casos y también con el conjunto de datos de prueba. De estos, vale la pena destacar los valores con el conjunto de datos de prueba: pérdida de **11.11** y RMSE de **3.40 °C**. Finalmente, en la Figura 6.56 se muestra un ejemplo de las predicciones que realiza el modelo (con la variable de temperatura de salida en la posición 60).

Caso	Pérdida	RMSE	Pérdida (validación)	RMSE (validación)
Caso 2	9.32	3.06	9.09	3.02
Caso 3	9.25	3.05	9.11	3.02
Caso 4	9.29	3.06	9.22	3.04
Caso 5	9.20	3.04	9.14	3.03
Prueba	11.11	3.40	N/A	N/A

Tabla 6.9: Resultados de modelo RNN con capa LSTM.

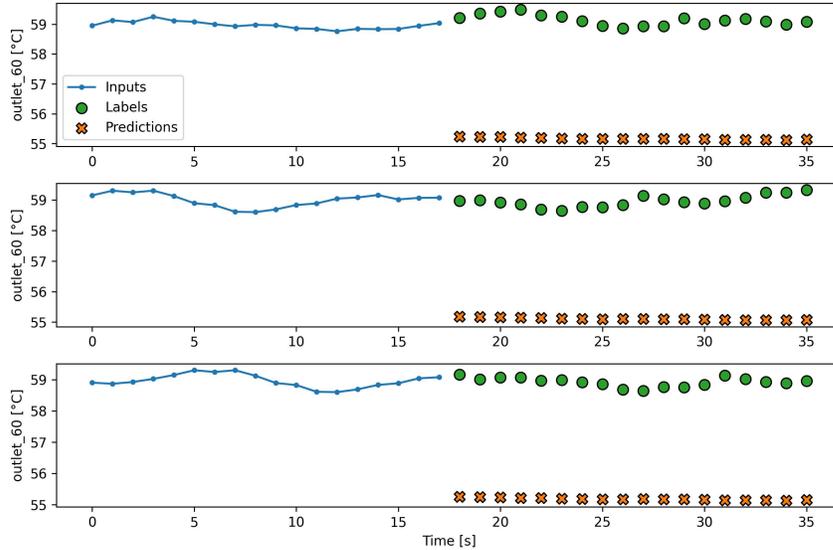


Figura 6.56: Predicciones de modelo RNN con capa LSTM.

6.7. Modelo autorregresivo

Se creó un modelo RNN autorregresivo. Este difiere de los modelos anteriores en que no utiliza las variables de velocidad de ventilador de entrada y de salida. Debido a la naturaleza de los modelos autorregresivos, este modelo utiliza todas las variables de temperatura (entrada y salida) como entrada y salida del modelo. De igual forma que los otros modelos, predice 18 observaciones en el futuro con 18 observaciones de entrada. Este consiste de una capa LSTM (con 32 unidades), una capa de *dropout* (con valor de 0.3) y la capa densa de salida. Este modelo utiliza las primeras 18 observaciones para hacer un “calentamiento”, para luego predecir y alimentar las salidas a los pasos posteriores (las 18 observaciones de salida). Se utilizó el optimizador Adam, el error cuadrático medio como función de pérdida y se entrenó durante 30 épocas.

6.7.1. Resultados

Este modelo tuvo un tiempo de entrenamiento total de **4 min 33.53 s**. A continuación, en la Tabla [6.10](#), se muestra el tiempo de entrenamiento por caso con la red lineal.

Caso	Tiempo
Caso 2	36.66 s
Caso 3	38.23 s
Caso 4	63.84 s
Caso 5	134.80 s

Tabla 6.10: Tiempo de entrenamiento por caso para el modelo con capas densas.

De igual forma, en la Tabla [6.11](#), se muestra los resultados obtenidos en los diferentes casos y también con el conjunto de datos de prueba. De estos, vale la pena destacar los valores con el conjunto de datos de prueba: pérdida de **11.54** y RMSE de **3.49 °C**. Finalmente, en la Figura [6.57](#)

se muestra un ejemplo de las predicciones que realiza el modelo (con la variable de temperatura de salida en la posición 60).

Caso	Pérdida	RMSE	Pérdida (validación)	RMSE (validación)
Caso 2	9.62	3.13	9.16	3.03
Caso 3	9.28	3.05	9.10	3.02
Caso 4	9.27	3.05	9.20	3.04
Caso 5	9.20	3.04	9.21	3.04
Prueba	11.54	3.49	N/A	N/A

Tabla 6.11: Resultados de modelo RNN autorregresivo .

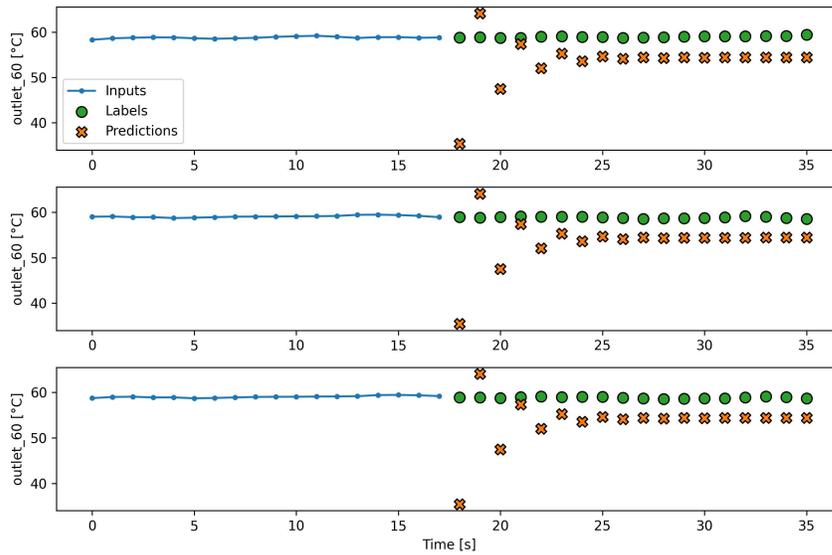


Figura 6.57: Predicciones de modelo RNN autorregresivo.

6.8. Comparación entre modelos

A continuación, en la Figura [6.58](#), se muestra la comparación entre los valores de RMSE obtenidos con el conjunto de datos de prueba para los diferentes modelos. Todos los modelos convergen a un valor de RMSE de aproximadamente 3 °C. El hecho que todos los modelos converjan prácticamente al mismo valor puede ser consecuencia del tamaño del conjunto de datos. Esto debido a que esto influyó directamente en el tamaño de la ventana (qué tantas observaciones se predicen en el futuro), lo que influye en la complejidad del problema que tienen que resolver los modelos y si la complejidad que aportan algunos modelos representa una mejora o no. Sin embargo, llamó mucho la atención que ninguno de los modelos fuera capaz de superar un valor de RMSE de 3. Incluso al hacer pruebas con más épocas, el valor de 3 representaba un asíntota en los resultados. Esto puede significar que este valor representa el valor mínimo global del error en el conjunto de datos o un mínimo local del cuál ninguno de los modelos fue capaz de salir.

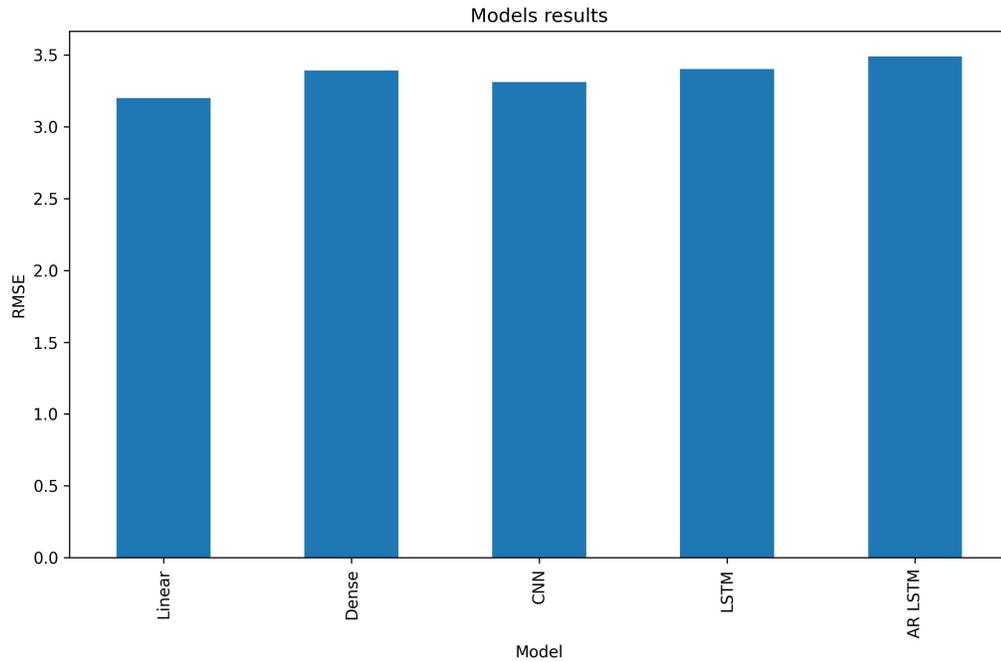


Figura 6.58: Comparación entre resultados de modelos

De igual forma, en la Figura [6.59](#) se muestra la comparación entre los tiempos de entrenamiento total (en segundos) de los diferentes modelos. En este caso, sí se puede apreciar una diferencia considerable entre los modelos. El modelo lineal y el de red neuronal convolucional tiene los tiempos de entrenamiento menores, seguidos por el de red neuronal con capas densas y el LSTM. Por último, se puede apreciar que el tiempo de entrenamiento del modelo autorregresivo RNN tiene un tiempo de entrenamiento considerablemente mayor en comparación a los otros modelos. Esto significa que el modelo de red neuronal convolucional tiene mucho potencial para este problema, puesto que es un modelo que es capaz de capturar la adyacencia temporal de los datos de la serie de tiempo y también es capaz de entrenar en un tiempo menor en comparación a los otros modelos. Sin embargo, es difícil distinguir si este modelo puede llegar a superar a los otros en términos de error con un conjunto de datos mayor.

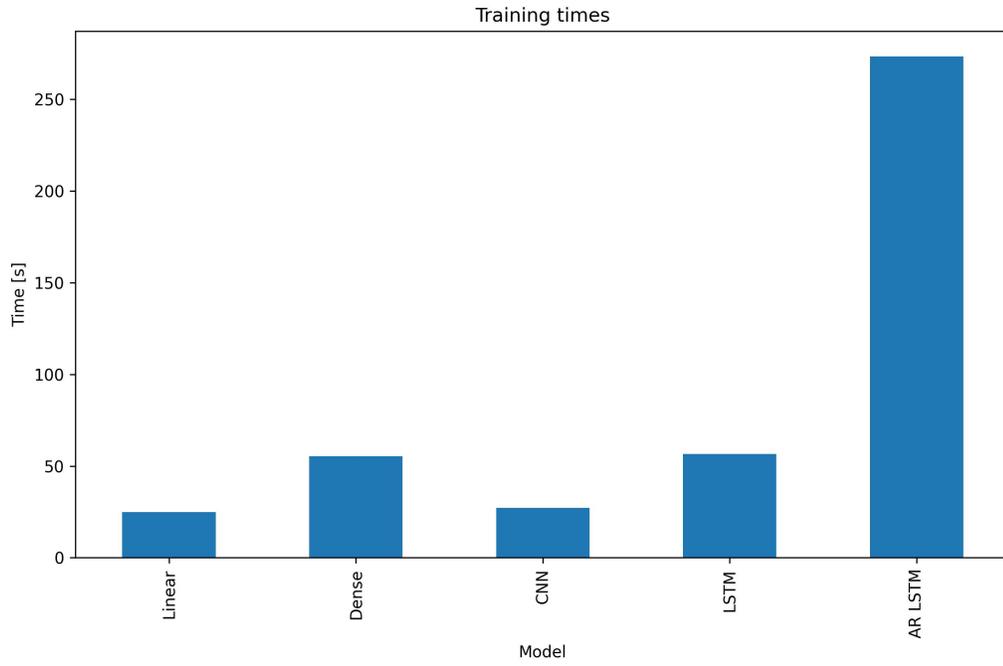


Figura 6.59: Comparación entre tiempo de entrenamiento de modelos

Luego de analizar los resultados obtenidos, se llegó a las siguientes conclusiones:

- Todos los modelos propuestos convergen en un valor aproximado de RMSE de 3.
- El valor de RMSE de 3 representa una asíntota en el entrenamiento para todos los modelos presentados, a pesar de realizar pruebas con diferentes combinaciones de hiperparámetros.
- A pesar de tener resultados prácticamente idénticos, la forma en la que los modelos predicen sí presentan diferencias (como se puede apreciar en las figuras de predicciones en los resultados de los modelos).
- El modelo lineal y el de red neuronal convolucional son los que menos tiempo de entrenamiento requirieron.
- El modelo autorregresivo RNN fue el modelo con mayor tiempo de entrenamiento.
- Los modelos con mayor complejidad no representaron una mejora en el rendimiento.
- El modelo de red neuronal convolucional muestra potencial debido al tiempo de entrenamiento y resultados obtenidos.
- La convergencia de los modelos puede ser consecuencia del tamaño del conjunto de datos.
- Se lograron los objetivos propuestos en el trabajo.

Recomendaciones

Luego de la realización del trabajo se sugieren las siguientes recomendaciones para futuros trabajos:

- Evaluar los modelos con un conjunto de datos de mayor tamaño, ya sea con datos de simulación CFD o datos experimentales.
- Evaluar los modelos con variables diferentes.
- Si se logra conseguir un conjunto de datos de mayor tamaño, evaluar el rendimiento de los modelos con mayor tamaño en las ventanas. Esto significa predecir más observaciones en el futuro y aumentar la complejidad del problema para los modelos.
- Realizar pruebas con mayor variedad de modelos, incluyendo modelos tradicionales de series de tiempo como lo puede ser un Vector AutoRegressive (VAR) o uno basado en árboles (p. ej. XGBoost). Esto con el fin de validar si se pueden obtener mejores resultados con el conjunto de datos.
- Aplicar modelos a nivel de instalación o incluso a nivel de gabinete.

-
-
- [1] Y. Futpagare, A. Joshi y A. Barghav, “Rack Level Forecasting Model of Data Center”, *16th IEEE ITherm Conference*, págs. 824-829, 2017.
 - [2] Y. Futpagare, A. Barghav y A. Joshi, “Predictive Model Development and Validation for Raised Floor Plenum Data Center”, *Journal of Electronic Packaging*, vol. 142, págs. 824-829, 2020.
 - [3] A. Saiyad, A. Patel, Y. Futpagare y A. Barghav, “Predictive modeling of thermal parameters inside the raised floor plenum data center using Artificial Neural Networks”, *Journal of Electronic Packaging*, vol. 42, 2021.
 - [4] IBM, *General guidelines for data centers*, 2021. dirección: <https://www.ibm.com/docs/en/psww2700/2.3.2.0?topic=determination-general-guidelines-data-centers>.
 - [5] S. R. y P. Norvig, *Artificial Intelligence: A Modern Approach*. Pearson, 2022.
 - [6] I. C. Education, *What is deep learning?* Dirección: <https://www.ibm.com/cloud/learn/deep-learning>.
 - [7] *Recurrent neural networks (RNN) with Keras* ; ; *Tensorflow Core*, agosto de 2022. dirección: <https://www.tensorflow.org/guide/keras/rnn>.
 - [8] J. Brownlee, *Long Short-Term Memory Networks With Python: Develop Sequence Prediction Models with Deep Learning*. Machine Learning Mastery, 2017. dirección: <https://books.google.com.gt/books?id=m7SoDwAAQBAJ>.
 - [9] R. Hyndman y G. Athanasopoulos, *Forecasting: principles and practice*. OTexts: Melbourne, Australia, 2018, [OTexts.com/fpp2](https://otexts.com/fpp2).
 - [10] Tensorflow, *Time Series forecasting* ; ; *Tensorflow Core*, 2022. dirección: https://www.tensorflow.org/tutorials/structured_data/time_series.
 - [11] Cisco, *What is a Data Center?*, CiscoEditor, ed., oct. de 2021. dirección: <https://www.cisco.com/c/en/us/solutions/data-center-virtualization/what-is-a-data-center.html>.
 - [12] *Colocation Data Centers*, 2022. dirección: <https://www.datacentermap.com/datacenters.html>.