Universidad Nacional de Loja

Facultad de la Energía, las Industrias y los Recursos Naturales No Renovables

Carrera de Electromecánica

Transfer learning para la detección de fallas en los aerogeneradores del parque Eólico Villonaco Loja-Ecuador.

Trabajo de Integración Curricular, previo a la obtención del título de Ingeniero Electromecánico.

AUTOR:

Freddy Santiago Morocho Cuenca

DIRECTOR:

Ing. Jorge Luis Maldonado Correa. PhD

Loja-Ecuador

2024

Certificado

Loja, 10 de diciembre de 2024

Ing. Jorge Luis Maldonado Correa. PhD
DIRECTOR DE TRABAJO DE INTEGRACIÓN CURRICULAR

CERTIFICO:

Que he revisado y orientado todo proceso de la elaboración del trabajo de Integración Curricular titulado: **Transfer learning para la detección de fallas en los aerogeneradores del parque Eólico Villonaco Loja-Ecuador** de autoría del estudiante **Freddy Santiago Morocho Cuenca**, con cédula de identidad **Nro. 1150233854** previa a la obtención del título de **INGENIERO ELECTROMECÁNICO**. Una vez que el trabajo cumple con todos los requisitos exigidos por la Universidad Nacional de Loja para el efecto, autorizo la presentación para la respectiva sustentación y defensa.

Firmado electrónicamente por:
JORGE LUIS
MALDONADO CORREA

Ing. Jorge Luis Maldonado Correa. PhD
DIRECTOR DEL TRABAJO DE
TITULACIÓN

Autoría

Yo, **Freddy Santiago Morocho Cuenca**, declaro ser autor del presente Trabajo de Integración Curricular y eximo expresamente a la Universidad Nacional de Loja y a sus representantes jurídicos de posibles reclamos y acciones legales, por el contenido del mismo. Adicionalmente acepto y autorizo a la Universidad Nacional de Loja la publicación de mí del trabajo de integración curricular o de titulación en el Repositorio Digital Institucional – Biblioteca Virtual.

Firma: 

Cédula de Identidad:1150233854

Fecha: 10 de diciembre de 2024

Correo electrónico: freddy.morocho@unl.edu.ec

Celular: 0962739607

Carta de autorización por parte del autor, para consulta, reproducción parcial o total y/o publicación electrónica del texto completo del Trabajo de Integración Curricular.

Yo **Freddy Santiago Morocho Cuenca** declaro ser autor del Trabajo de Integración Curricular titulado **Transfer learning para la detección de fallas en los aerogeneradores del parque Eólico Villonaco Loja-Ecuador** como requisito para optar el título de **Ingeniero Electromecánico** autorizo al sistema Bibliotecario de la Universidad Nacional de Loja para que con fines académicos muestre la producción intelectual de la Universidad, a través de la visibilidad de su contenido de la siguiente manera en el Repositorio Institucional.

Los usuarios pueden consultar el contenido de este trabajo en el Repositorio Institucional, en las redes de información del país y del exterior con las cuales tenga convenio la Universidad.

La Universidad Nacional de Loja, no se responsabiliza por el plagio o copia del trabajo de integración curricular que realice un tercero.

Para constancia de esta autorización, en la ciudad de Loja, a los diez días del mes de diciembre del dos mil veinticuatro.

Firma: 

Autor: Freddy Santiago Morocho Cuenca
Cédula: 1150233854
Dirección: Samana
Correo electrónico: freddy.morocho@unl.edu.ec
Celular:0962739607

DATOS COMPLEMENTARIOS:
Director del trabajo de integración curricular o de titulación: Ing. Jorge Luis Maldonado Correa. PhD.

Dedicatoria

Con un profundo amor, dedico el presente trabajo a mis amados padres, Saul Morocho y Maria Cuenca su amor incondicional han sido el faro que a iluminado mi camino, sus consejos y su determinación inquebrantable son fuentes inagotables de inspiración, a ellos les dedico este triunfo, como expresión de gratitud por todo lo que han contribuido en mi vida, y como un testimonio de que su apoyo es la fuerza motriz detrás de los éxitos alcanzados y de aquellos que están por venir.

A mis queridos hermanos, Rober y Juan, quienes son compañeros inseparables en mis momentos de mayor alegría y dificultades, a lo largo de la vida, han sido mi sostén inquebrantable, brindándome su apoyo incondicional, su motivación constante y, sobre todo, su inmenso amor. Agradezco profundamente la dicha de contar con su presencia, amistad y hermandad en cada capítulo de mi historia.

Freddy Santiago Morocho Cuenca

Agradecimientos

Expreso mi agradecimiento a mis padres por su inquebrantable respaldo, tanto económico como moral, a lo largo de mi formación, su generosidad y apoyo han sido el pilar que ha permitido mi crecimiento y desarrollo académico.

Expreso mi sincero agradecimiento al Ingeniero Jorge Luis Maldonado Correa, quien ha desempeñado un papel fundamental en la realización de este trabajo, el tiempo dedicado, su confianza desde el inicio de este proceso han sido invaluables, su apoyo y orientación han dejado una huella significativa en mi desarrollo académico y profesional.

A mis tíos, Alfredo Cuenca, Gabriel Cuenca y Néstor Cuenca, a quienes agradezco sinceramente por ser una fuente constante de motivación en mi trayectoria académica su aliento y su inquebrantable apoyo me han impulsado a seguir adelante y nunca rendirme ante los desafíos. A mi familia en su totalidad, quienes siempre han compartido valiosos consejos, les dedico mi más profundo agradecimiento.

A Karina Calderon quien ha sido un pilar fundamental en mi trayecto universitario, brindándome su apoyo incondicional a lo largo de todo este proceso, a mis compañeros y amigos que compartimos toda la experiencia de vivir la vida universitaria.

A la Universidad Nacional de Loja y a todos los docentes por brindarme la invaluable oportunidad de acceder al mundo profesional, permitiéndome no solo adquirir conocimientos académicos, sino también compartiendo sus saberes personales y profesionales.

Freddy Santiago Morocho Cuenca

Índice de contenidos

PORTADA . . . . . **i**

CERTIFICACIÓN . . . . . **ii**

AUTORÍA . . . . . **iii**

CARTA DE AUTORIZACIÓN . . . . . **iv**

DEDICATORIA . . . . . **v**

AGRADECIMIENTO . . . . . **vi**

ÍNDICE DE CONTENIDO . . . . . **vii**

Índice de tablas xi

Índice de figuras xiii

Índice de Anexos xiv

Índice de Ecuaciones xv

SIMBOLOGÍA . . . . . **xvi**

1 TÍTULO **1**

2 RESUMEN **2**

3 INTRODUCCIÓN **4**

4 MARCO TEÓRICO **6**

4.1 Capítulo 1 6

4.1.1 Energía Eólica 6

4.1.2 Central Eólica Villonaco (CEV) 6

4.1.3 Aerogeneradores 7

4.1.4 Sistema SCADA 10

4.1.4.1 Arquitectura SCADA. 10

4.1.5 Operación y Mantenimiento (O&M) de Aerogeneradores 11

4.1.5.1 Tipos de mantenimiento. 11

4.1.5.1.1 Mantenimiento Preventivo y Mantenimiento Correctivo. 12

4.1.5.1.2 Mantenimiento Mejorativo. 12
4.1.5.1.3 Mantenimiento Programado y Mantenimiento No programado. 12
4.2 Capítulo 2 12
4.2.1 Data Science 12
4.2.2 Data Mining 13
4.2.3 Correlación 13
4.2.3.1 Matriz de correlación. 14
4.2.3.2 Factor de inflación de la varianza (VIF). 14
4.2.4 Data Sampling 15
4.2.4.1 Sobremuestreo. 15
4.2.4.2 Submuestreo. 15
4.2.4.3 Synthetic Minority Oversampling Technique (SMOTE) 16
4.2.5 Normalización 16
4.2.5.1 Normalización z o estandarización. 16
4.2.5.2 Normalización min-máx. 17
4.3 Capítulo 3 17
4.3.1 Deep Learning (DL) 17
4.3.1.1 Red Neuronal Artificial (ANN). 18
4.3.1.2 Multilayer Perceptron (MLP). 19
4.3.1.3 Recurrent Neural Network (RNN). 20
4.3.1.3.1 Tipos de RNN. 21
4.3.1.4 Long Short Term Memory (LSTM). 22
4.3.1.5 Gated-Recurrent-Units (GRU). 23
4.3.1.6 Transfer learning (TL) 24
4.3.2 Overfitting y Underfitting 24
4.3.2.1 Overfitting o Sobreajuste. 24
4.3.2.2 Underfitting o Subajuste. 24
4.3.3 Métricas de Evaluación 25
4.3.3.1 Accuracy o Exactitud. 26
4.3.3.2 Recall o sensibilidad. 26
4.3.3.3 Specifity o Especificidad. 26
4.3.3.4 Precision o Precisión. 26

4.3.3.5 F1-score. 26
4.3.3.6 Curva ROC. 27
4.4 Capítulo 4 27
4.4.1 Interfaz gráfica de usuario GUI 27
4.4.1.1 Dashboard. 28

5 METODOLOGÍA 29
5.1 Área de trabajo 29
5.1.1 Localización 29
5.2 Equipos y materiales 30
5.2.1 Equipos 30
5.2.2 Materiales 30
5.2.2.1 Base de Datos. 30
5.2.2.2 Python. 30
5.2.2.3 Latex. 31
5.3 Procedimiento 31
5.3.1 OE1. Implementar Data Mining al registro histórico del periodo 2014-2021 de los datos SCADA y el registro de alarmas de fallos del parque eólico Villonaco del cantón Loja 32
5.3.2 OE2. Aplicar transfer learning para predicción de fallas y evaluar su rendimiento mediante diferentes métricas 34
5.3.3 OE3. Desarrollar un dashboard para el manejo y visualización de los resultados obtenidos 36
5.4 Procesamiento y análisis de datos 36
5.4.1 OE1. Implementar Data Mining al registro histórico del periodo 2014-2021 de los datos SCADA y el registro de alarmas de fallos del parque eólico Villonaco del cantón Loja 36
5.4.1.1 Base de Datos. 36
5.4.1.2 Registro de alarmas. 38
5.4.1.3 Etiquetado de datos. 38
5.4.1.4 Manejo de datos faltantes. 39
5.4.1.5 Selección de variables a trabajar. 39

5.4.2 OE2. Aplicar transfer learning para predicción de fallas y evaluar su rendimiento mediante diferentes métricas . . . 40
5.4.2.1 Divisón de regiones A,B y C . . . 40
5.4.2.2 Remuestreo. . . . 41
5.4.2.3 Tratamiento preliminar para entrenar las redes neuronales. . . . 41
5.4.2.4 Aplicación de deep learning. . . . 42
5.4.2.4.1 RNN, LSTM y GRU. . . . 42
5.4.2.5 Metrícas de evaluación de desempeño. . . . 44
5.4.2.6 Transfer learning (TL). . . . 45
5.4.2.6.1 Transfer Learning 1 (TL1). . . . 47
5.4.2.6.2 Transfer Learning 2 (TL2). . . . 47
5.4.3 OE3. Desarrollar un dashboard para el manejo y visualización de los resultados obtenidos . . . 49

6 RESULTADOS . . . 50
6.1 Implementar Data Mining al registro histórico del periodo 2014-2021 de los datos SCADA y el registro de alarmas de fallos del parque eólico Villonaco del cantón Loja . . . 50
6.2 Aplicar transfer learning para predicción de fallas y evaluar su rendimiento mediante diferentes métricas . . . 54
6.2.1 Transfer learning 1 (TL1) . . . 54
6.2.2 Transfer learning 2 (TL2) . . . 60
6.3 Desarrollar un dashboard para el manejo y visualización de los resultados obtenidos . . . 64

7 DISCUSIÓN . . . 67

8 CONCLUSIONES . . . 70

9 RECOMENDACIONES . . . 72

10 BIBLIOGRAFÍA . . . 73

11 ANEXOS . . . 78

Índice de tablas

Tabla 1. Datos técnicos Goldwind GW70/1500. 8

Tabla 2. Interpretación del valor VIF dependiendo la multicolinealidad. 15

Tabla 3. Matriz de confusión binaria. 25

Tabla 4. Muestra de datos entregados por CELEC EP. 37

Tabla 5. Muestra del registro de alarmas entregados por CELEC EP. 38

Tabla 6. Indicadores estadística de algunas variables de estudio. 50

Tabla 7. Variables seleccionadas para entrenar el modelo base y evaluar los diferentes estrategias de TL. 54

Tabla 8. Configuración de hiperparámetros de las redes RNN, LSTM y GRU. . . . 56

Tabla 9. Métricas de evaluación los modelos utilizados en el TL1. 57

Tabla 10. Resultados del TL2 sin reajuste de hiperparámetros. 62

Tabla 11. Resultados del TL2 con reajuste de hiperparámetros. 63

Índice de figuras

Figura 1. Central Eólica Villonaco. 7
Figura 2. Componentes del aerogenerador GW70/1500. 10
Figura 3. Estructura sistema SCADA GOLDWIND. 11
Figura 4. Proceso de data mining. 13
Figura 5. Ejemplo matriz de correlación. 14
Figura 6. Balanceo de Datos, Undersampling y Oversampling. 16
Figura 7. Clasificación de la Inteligencia Artificial. 18
Figura 8. (a) Neurona, (b) Perceptrón. 19
Figura 9. Arquitectura de una MLP. 20
Figura 10. Estructura red neuronal recurrente. 21
Figura 11. Tipos de RNN con base en su longitud de entrada y salida. 22
Figura 12. Diagrama de una unidad LSTM. 22
Figura 13. Diagrama de una unidad GRU. 23
Figura 14. Sobreajuste y subajuste de un modelo. 25
Figura 15. Curva ROC. 27
Figura 16. Capas de interfaz gráfica de usuario. 28
Figura 17. Ejemplo dashboard. 28
Figura 18. Ubicación CEV. 29
Figura 19. Flujograma para el desarrollo del trabajo de titulación. 32
Figura 20. Estructura de los datos entregados por CELEC EP. 33
Figura 21. Fragmento de base de datos y muestra de selección de regiones A-B-C. . 35
Figura 22. Base de datos por WT. (a) Archivos de cada WT en formato .cvs, (b) Datos WT1 con fecha (timestamp) en orden ascendente. 37
Figura 23. Etiquetado de base de datos para determinar el WT con mayor número de fallas y alarmas. 39
Figura 24. División de regiones para el modelo base y TL1 41
Figura 25. Paquetes para cargar datos en las redes neuronales y para crear la red neuronal. 42
Figura 26. Código Python de red neuronal RNN, con la biblioteca *PyTorch*. 43
Figura 27. Código Python de red neuronal LSTM, con la biblioteca *PyTorch*. 43
Figura 28. Código Python de red neuronal GRU, con la biblioteca *PyTorch* 44

Figura 29. Código Python para el cálculo de las diferentes métricas. 45
Figura 30. Estrategia 1 de TL. 46
Figura 31. Estrategia 2 de TL. 46
Figura 32. Estrategia 3 de TL. 47
Figura 33. Retiquetado de datos, solo cuando existe un paro por fallo de WT. 48
Figura 34. Importación de paquetes dash, plotly y html, para el desarrollo del dasboard. 49
Figura 35. Distribución de datos durante los 7 años variable wind_speed_avg. 51
Figura 36. Muestra datos "NaN" para el WT5. 52
Figura 37. Cantidad de datos etiquetados de las WTs. 53
Figura 38. Curva distribuición wind_speed_max, label vs tiempo. 55
Figura 39. Frecuencia de etiquetas en las regiones A-B-C. 55
Figura 40. Resultados de algoritmo de oversampling. a) No oversampling, b) Oversampling, c) Total etiquetas. 56
Figura 41. Curvas de las funciones de pérdida. 57
Figura 42. Conjunto de curvas ROC. 58
Figura 43. Resultados de aplicar el TL1. 59
Figura 44. Total de etiquetas para cada WT. 61
Figura 45. Ventana Home del dashboard. 64
Figura 46. Ventana data mining del dashboard. 65
Figura 47. Ventana TL1 del dashboard. 66
Figura 48. Ventana TL2 del dashboard. 66

Índice de Anexos

Anexo 1. Código dashboard 78
Anexo 2. Manual de usuario 78
Anexo 3. Certificado de traducción abstract 79

Índice de Ecuaciones

Factor de inflación de la varianza	$VIF_i = \frac{1}{1-R_i^2}$	14
Normalización z	$\hat{X}[:,i] = \frac{X[:,i]-\mu_i}{\sigma_i}$	16
Media arimética	$\mu_i = \frac{1}{N}\sum_{k=1}^{N} X[k,i]$	17
Desviación estándar	$\sigma_i = \sqrt{\frac{1}{N-1}\sum_{k=1}^{N}(X[k,i]-\mu_i)^2}$	17
Normalización min-máx	$\hat{X}[:,i] = \frac{X[:,i]-min(X[:,i])}{max(X[:,i])-min(X[:,i])}$	17
Perceptrón	$y = f(\sum_{i=1}^{n} x_i \cdot w_i + b)$	19
RNN	$h^{(t)} = \tanh(Ux^{(t)} + Wh^{(t-1)} + b)$	21
	$\hat{y} = softmax(Vh^{(t)} + c)$	21
LSTM	$i^t = \sigma(W^i x^t + U^i h^{t-1} + b^i)$	23
	$j^t = g(W^j x^t + U^j h^{t-1} + b^j)$	23
	$f^t = \sigma(W^f x^t + U^f h^{t-1} + b^f)$	23
	$s^t = f^t \otimes s^{t-1} + i^t \otimes j^t$	23
	$o^t = \sigma(W^o x^t + U^o h^{t-1} + b^o)$	23
	$h^t = o^t \otimes g(s^t)$	23
GRU	$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z)$	24
	$r_t = g(W_r x_t + U_r h_{t-1} + b_r)$	24
	$\tilde{h_t} = tanh(W_h x_t + U_r(r_t \otimes h_{t-1}))$	24
	$h_t = (1 - z_t) \otimes h_{t-1} + z_t \otimes \tilde{h_t}$	24
Accuracy	$Accuracy = \frac{VP+VN}{VP+VN+FN+FP}$	26
Reccall	$Recall = \frac{VP}{VP+FN}$	26
Specifity	$Specifity = \frac{VN}{VN+FP}$	26
Precision	$Precision = \frac{VP}{VP+FP}$	26
F1-score	$F1 - score = \frac{2 \times Precision \times Recall}{Precision+Recall}$	26

Simbología

Acrónimos

CEV	Central eólica Villonaco.
SCADA	Supervisory Control and Data Acquisition.
O&M	Operación y Mantenimiento.
WT	Turbina eólica (Wind Turbine).
WTs	Turbinas eólicas.
DD	Direct Drive.
IA	Inteligencia artificial.
DL	Deep Learning.
ANN	Red neuronal artificial.
RNN	Recurrent Neural Network.
LSTM	Long Short-Term Memory.
GRU	Gated recurrent unit.
TL	Transfer learning.
OE	Objetivo específico.
CELEC EP	Empresa Pública Corporación Eléctrica del Ecuador.
VIF	Factor de inflación de la varianza.
SMOTE	Synthetic Minority Over-sampling Technique.
MLP	Multilayer percentron.
GUI	Interfaz gráfica de usuario.
OE1	Objetivo específico 1.
OE2	Objetivo específico 2.
OE3	Objetivo específico 3.
NaN	Not a Number.
PCA	Análisis de componentes principales.

1. Título

Transfer learning para la detección de fallas en los aerogeneradores del parque Eólico Villonaco Loja-Ecuador.

2. Resumen

El presente trabajo de titulación aborda la detección de fallos en aerogeneradores de la Central Eólica Villonaco (CEV). Para ello se propone el uso de Transfer Learning por ser una técnica innovadora y de alta precisión en tareas de predicción.

En este sentido, en primer lugar se aplica un proceso de data mining a los datos históricos del sistema SCADA y a los registro de fallos comprendidos entre 2014-2021. Luego se crea un modelo base con el aerogenerador más propenso a fallos con ayuda de las redes RNN, LSTM y GRU.

A continuación, se aplica Transfer Learning a los diferentes aerogeneradores como al mismo aerogenerador que se utilizó para el modelo base. Finalmente se utilizan diferentes métricas para evaluar el rendimiento del modelo tanto con datos balanceados como desbalanceados, y se presentan los resultados en un dashboard interactivo.

Entre los resultados se puede destacar la precisión del Transfer Learning para anticipar fallos en el aerogenerador, y su capacidad para adaptarse a diferentes aerogeneradores, logrando obtener resultados prometedores en datos desbalanceados. La introducción del Transfer Learning se muestra como una estrategia potente y eficaz en el desarrollo de inteligencias artificiales para el mantenimiento predictivo, lo que redundará en beneficios potenciales para la CEV y para otros parques eólicos en Ecuador.

Palabras claves: Transfer Learning, Central Eólica Villonaco, SCADA, RNN, LSTM, GRU

Abstract

This degree work deals with detecting failures in the Villonaco Wind Farm (VWF) wind turbines. For this purpose, Transfer Learning is proposed as an innovative and highly accurate technique for prediction tasks.

In this sense, first, a data mining process is applied to the SCADA system's historical data and failure records from 2014-2021. Then, with the help of the RNN, LSTM, and GRU networks, a base model with the most failure-prone wind turbine is created.

Transfer Learning is then applied to the different wind turbines and to the same wind turbine that was used for the base model. Finally, different metrics are used to evaluate the model's performance with both balanced and imbalanced data, and the results are presented in an interactive dashboard.

Among the results, we can highlight the accuracy of Transfer Learning in anticipating wind turbine failures and its ability to adapt to different wind turbines, achieving promising results in unbalanced data. The introduction of Transfer Learning is a powerful and effective strategy for developing artificial intelligence for predictive maintenance, which will result in potential benefits for the VWF and other wind farms in Ecuador.

3. Introducción

En la provincia de Loja-Ecuador se encuentra ubicada la Central Eólica Villonaco (CEV) en el cerro Villonaco, que cuenta con 11 aerogeneradores, cada uno con una potencia nominal de 1,5MW y desarrollan una potencia nominal total de 16,5MW (Instituto Nacional De Eficiencia Energética y Energías Renovables INER, 2014), debido a que los ingresos de la central depende de la producción de energía, los fallos en los aerogeneradores (WTs) pueden reducir la producción y generar costos de operación y mantenimiento (O&M) por lo que muchos investigadores desarrollan métodos para la detección temprana de fallos.

Partiendo de esté contexto el presente trabajo titulado "Transfer learning para la detección de fallas en los aerogeneradores del parque Eólico Villonaco Loja-Ecuador", se justifica ante la necesidad de investigar los diferentes métodos para la detección temprana de fallos.

Con estas premisas, resulta de especial interés utilizar herramientas de Data Science para la detección de fallas en los aerogeneradores utilizando los datos operativos y de alarmas del sistema Supervisory Control and Data Acquisition (SCADA), e implementar una interfaz gráfica para la visualización de los datos y resultados, simultáneamente se tiene en cuenta que las investigaciones sobre centrales eólicas y aerogeneradores han tenido un gran desarrollo, en el artículo titulado "Scientometric review of artificial intelligence for operations & maintenance of wind turbines: The past, present and future", realiza una revisión cienciométrica sobre la evolución conceptual y temática de la inteligencia artificial (IA) en el sector de la energía eólica, de esta manera Chatterjeea y Dethlefs (2022) determina que la mayoría de investigaciones están orientados al diagnostico y detección de fallos.

La investigación incorpora un enfoque innovador de Transfer Learning (TL), mediante la aplicación de un modelo previamente entrenado con algoritmos de Deep Learning (DL). Esta estrategia aprovecha los conocimientos adquiridos en un dominio relacionado para mejorar el rendimiento de la predicción de fallos en el contexto específico de la CEV. La aplicación de TL representa un paso significativo hacia la optimización de los recursos y la maximización de la eficacia en la detección de fallos, debido a que la CEV abastece el 25 % de demanda eléctrica de la provincia de Loja. La detección anticipada de fallos es crucial, ya que permite la toma de decisiones preventivas, evitando daños y paros prolongados, generando así mayor rentabilidad económica, eficiencia operativa y reducción de costos de O&M.

Es pertinente destacar que, en el contexto ecuatoriano, se están gestando nuevos proyectos eólicos, como Villonaco II, entre otros. Por ende, los resultados obtenidos en esta investiga-

ción no solo beneficiarán a la CEV, sino que también podrán aplicarse a otros parques y futuros proyectos eólicos, contribuyendo al avance y sostenibilidad del sector.

Con estos antecedentes, los objetivos de este trabajo de integración curricular son:

Objetivo general

Aplicar transfer learning para detección de fallas en los aerogeneradores utilizando los datos operativos y de alarmas del sistema SCADA, e implementar una interfaz gráfica para la visualización de los datos y resultados.

Objetivos específicos

- Implementar Data Mining al registro histórico del periodo 2014-2021 de los datos SCADA y el registro de alarmas de fallos del parque eólico Villonaco del cantón Loja.
- Aplicar transfer learning para predicción de fallas y evaluar su rendimiento mediante diferentes métricas.
- Desarrollar un dashboard para el manejo y visualización de los resultados obtenidos.

4. Marco teórico

4.1 Capítulo 1

El propósito de este capítulo es proporcionar información general sobre energía eólica y un acercamiento al funcionamiento de los aerogeneradores GOLDWIND ubicados en la CEV; además, se introduce el concepto de mantenimiento predictivo el cual, es el principal motivante de este trabajo.

4.1.1 Energía Eólica

La energía eólica es la energía que se origina del movimiento de las masas de aire (viento), los vientos son producto de la radiación electromagnética solar que produce un calentamiento no uniforme de la superficie terrestre, debido a esto en el día las masas de aire sobre la tierra se calientan y se elevaban por falta de densidad, mientras el aire más frío sobre el agua, océanos, mares y lagos se desplazan para ocupar el lugar, logrando de esta manera un flujo de aire. De toda la energía que la superficie terrestre recibe del sol, entre el 1 % y 2 % se convierte en viento (Alberto, 2013).

A pesar de existir otras fuentes de energías el cambio climático es un factor que a incentivado la expansión de la energía eólica, debido que es una fuente energética limpia, económica, competitiva y con un desarrollo tecnológico muy madura, además de ser una energía renovable (TECH4CDM, 2009).

4.1.2 Central Eólica Villonaco (CEV)

La central eólica inicio su construcción en agosto del 2011, desde entonces se a destacado por sus velocidades de viento promedio anual superior a 10 m/s, y por ser la primera central eólica en Ecuador continental, como su nombre lo indica se encuentra en el cerro Villonaco ubicado en la provincia de Loja, cantón Loja (Soto, 2017). En la **Figura 1** se observa parte de la central eólica Villonaco.

Figura 1. Central Eólica Villonaco.

La central se sitúa a 2720 msnm y contempla 11 aerogeneradores del tipo GW70/1500, a lo largo de la línea de 2 km aproximadamente, cada aerogenerador posee una potencia nominal de 1.5MW, logrando una potencia nominal total de generación por la central de 16.5 MW (Instituto Nacional De Eficiencia Energética y Energías Renovables INER, 2014).

4.1.3 Aerogeneradores

Una turbina eólica convierte el movimiento de masas de aire en energía mecánica y posteriormente en energía eléctrica, la primera conversión se da en las palas del rotor debido a que estás captan la energía del viento produciendo un movimiento rotacional, luego se produce la segunda transformación en el generador eléctrico que aprovecha el movimiento rotacional.

Como ya se mencionó, los aerogeneradores son de la marca GoldWind modelo GW70/1500 con potencia nominal de 1.5 MW de fabricación china, según GOLDWIND (2011), en su manual técnico, indica que el aerogenerador es de imán permanente, no posee engranajes y es de accionamiento directo por un rotor de 3 palas, además de contar con tecnología Direc Drive (DD) que reduce los componentes de la turbina y aumenta la fiabilidad alcanzado una máxima eficiencia.

En la **Tabla 1** se presentan las principales características del aerogenerador GW70/1500.

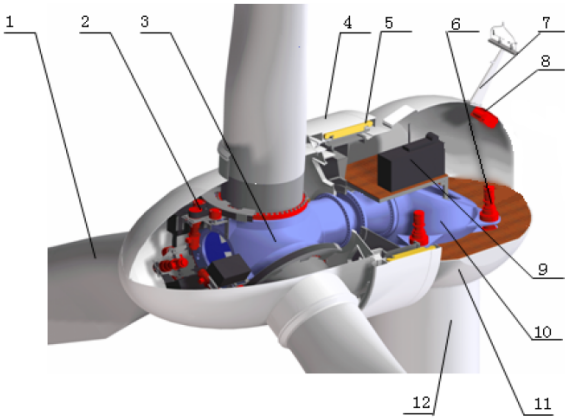
Tabla 1. Datos técnicos Goldwind GW70/1500.

Parámetros	Especificaciones
Potencia	
Potencia nominal	1500 kW
Velocidad de viento Cut-in	3 m/s
Velocidad nominal del viento	11.8 m/s
Velocidad de viento Cut-out	25 m/s
Rotor	
Diámetro	70.34 m
Área barrida	3886 m^2
Rango de velocidad	10.2∼19 rpm
Palas	3
Tipo	LM34P
Control	Collective Pitch Control / Rotor Speed Control
Sistema de seguridad	
	Control de Pitch de pala independiente Freno de disco hidráulico Bloqueo de rotor hidráulico
Generador	
Potencia nominal	1500 kW
Voltaje nominal	690 V
Corriente nominal	660 A
Diseño	Direct drive
Velocidad de rotación nominal	19 rpm
Clase de protección	IP23
Categoría de aislamiento	F
Tipo	Generador síncrono multipolar, imán permanente excitado
Sistema de orientación	
Concepto de diseño	motor de accionamiento eléctrico
Movimiento nominal	0.45^o/sec
Sistema de orientación	Freno 10 de retención
Torre	
Tipo	Torre de acero tubular

Parámetros	Especificaciones
Altura	65 m
Diseño estándar	IEC1024, cumple con GL estándar
Resistencia a tierra	$\leq 4\Omega$
Base	Base Plana
Convertidor	
Tipo	Sistema modular IGBT
Clase de protección	IP54
rango del factor de potencia de salida	-0.95∼+0.95
Voltaje nominal de salida	620/690 V
Corriente nominal de salida	1397/1255 A
Transformador	
Voltaje de entrada	620 V
Voltaje de salida	20 kV
Sistema de control	
Tipo	Microprocessor Controlled, DFÜ (SCADA) PLC

Fuente: GOLDWIND (2011).

Los aerogeneradores GW70/1500, internamente constan de los componentes que se muestran en la **Figura 2**. Es importante mencionar que este modelo no cuenta con caja reductora de velocidad y su traslado se dio mediante vía marítima hasta Puerto Bolivar y luego por vía terrestre hasta la ciudad de Loja.



1. Aspa 2. Sistema de control de paso 3. Cubo 4. Rotor del generador 5. Estátor del generador
6.Accionamiento de orientación 7. Sistema de medición de viento 8. Malacate 9. Caja superior de la góndola
10. Base del marco de la góndola 11. Góndola 12. Torre

Figura 2. Componentes del aerogenerador GW70/1500.

Fuente: GOLDWIND (2011).

4.1.4 Sistema SCADA

La palabra SCADA proviene de Supervisory Control and Data Acquisition o Control con Supervisión y Adquisición de Datos, y se trata de un software que permite el acceso a datos remotos de un proceso o varios, además el control de estos mediante herramientas de comunicación. SCADA hace la función de interfaz entre los niveles de control de baja jerarquía con las gestión de un nivel alto (Ramírez, 2022).

4.1.4.1 Arquitectura SCADA.

Según Vasconez (2019), el sistema SCADA se compone como mínimo de:

- Medición / Sensores.
- Comunicación de Datos
- Controladores.
- Servidores.
- Visualizadores.
- Dispositivos de Control.

En la CEV se cuenta con sistema SCADA GOLDWIND que trabaja como se muestra en la **Figura 3**, el cual permite monitorear externamente desde cualquier computador conectado a

Internet, donde se puede observar los datos operativos en tiempo real, fallos, datos históricos guardados cada 10 minutos, además generar informes de producción, gráficas de tendencia, entre otros.

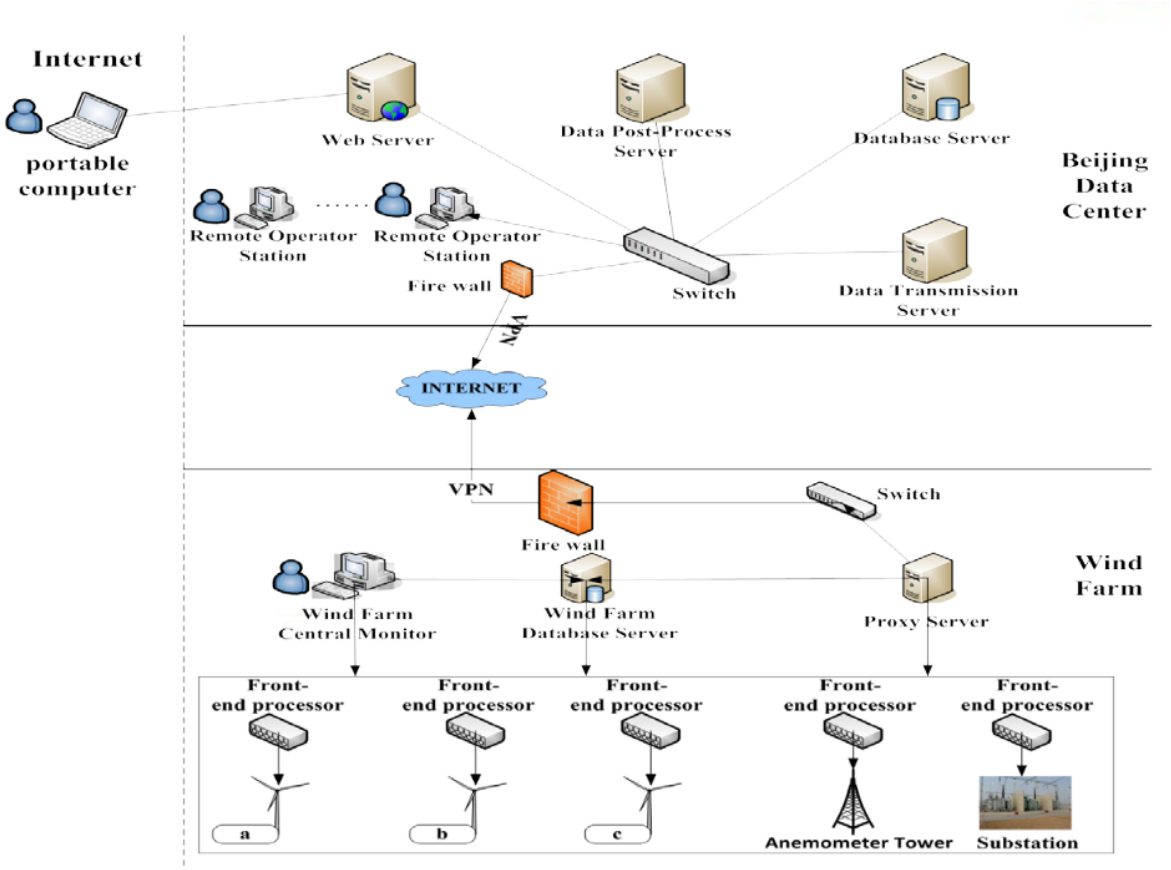


Figura 3. Estructura sistema SCADA GOLDWIND.

Fuente: GOLDWIND (2011).

4.1.5 Operación y Mantenimiento (O&M) de Aerogeneradores

El mantenimiento es el conjunto de acciones o actividades encaminadas a garantizar el correcto funcionamiento de las máquinas e instalaciones que conforman un proceso de producción permitiendo que este alcance su máximo rendimiento y costos mínimos (Olarte et al., 2010).

4.1.5.1 Tipos de mantenimiento.

Según Sexto (2018), concluye que los tipos de mantenimiento se simplifican principalmente en:

- Mantenimiento Preventivo y Mantenimiento Correctivo.
- Mantenimiento Mejorativo.
- Mantenimiento Programado y Mantenimiento No programado.

4.1.5.1.1 Mantenimiento Preventivo y Mantenimiento Correctivo.

Se busca mantener las funciones originales del activo y no existe cambios en las características de diseño. El mantenimiento predictivo recoge información en tiempo real sobre el estado de los componentes más sensibles de la máquina y, basándose en un histórico de datos vinculados con las averías más comunes, información sobre su funcionamiento actual y el uso de herramientas de inteligencia artificial para su análisis, es capaz de ofrecer una previsión de cuándo se va a estropear o es necesario cambiar un determinado componente. El grado más avanzado se denomina mantenimiento proactivo y, además de ofrecer una previsión de cuándo puede fallar algo, proporciona información sobre cuáles son los motivos por los que se producirá ese fallo. Nos indica las causas, lo que nos permite tomar las medidas más adecuadas para preservar los procesos de fabricación (Montero, 2020).

4.1.5.1.2 Mantenimiento Mejorativo.

Se busca realizar cambios en las características intrínsecas dadas por el diseño pero sin cambiar las funciones originales.

4.1.5.1.3 Mantenimiento Programado y Mantenimiento No programado.

Se realiza un análisis para asignar fechas, tiempos y recursos para ejecutar determinadas acciones de mantenimiento.

4.2 Capítulo 2

En este capítulo se aborda los temas referentes al procesamiento de los datos en los cuales se incluye: limpieza, selección de variables, entre otros, a partir de una base de datos brutos obtenidos del sistema SCADA.

4.2.1 Data Science

El Data Science abarca un conjunto de principios, definiciones de problemas, algoritmos y procesos para la extracción de patrones, haciendo uso de un gran volumen de datos. El Data Science se a desarrollado en campos relacionados, como la estadística, la minería de datos, el aprendizaje automático y el análisis predictivo (Kelleher & Tierney, 2018).

Según Lopez (2023), nos plantea el proceso que utiliza el Data Science para explorar el mundo de datos, los cuales son:

- Objetivo de Investigación.
- Obtención de datos.
- Preparación de datos.

- Exploración de datos.
- Construcción de modelos.
- Presentación de resultados y automatización de análisis.

De la misma manera el autor corrobora que no es necesario seguir un orden descendente de los puntos mencionados, dado que a menudo se debe iterar entre las diferentes etapas dependiendo los resultados que vayamos obteniendo.

4.2.2 Data Mining

Bello (2021), manifiesta que el data mining es un conjunto de técnicas y tecnologías que permiten explorar grandes bases de datos, de manera automática o semiautomática, con el objetivo de encontrar patrones repetitivos que expliquen el comportamiento de estos datos usando estadística o inteligencia artificial (IA).

El Data Mining se clasifica como una disciplina del Data Science. En la **Figura 4**, se muestran los pasos a llevar para un correcto minado de datos:

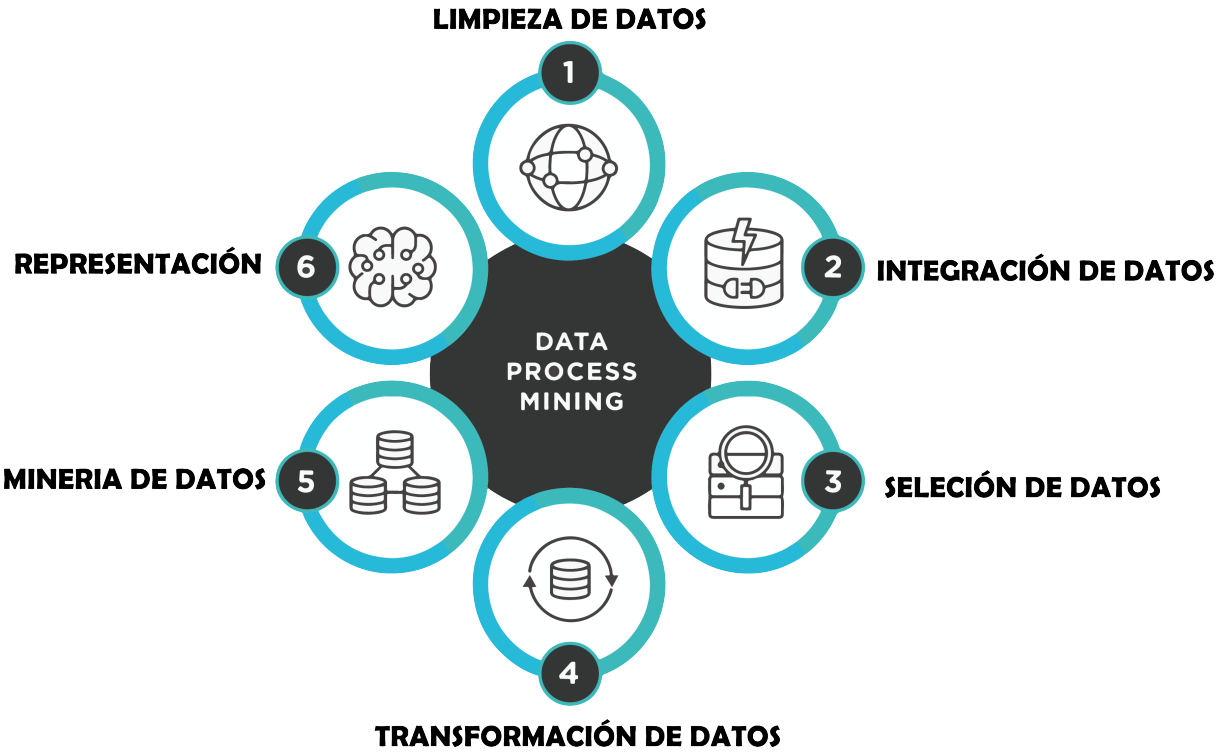


Figura 4. Proceso de data mining.

Fuente: spotfire (2024).

4.2.3 Correlación

La correlación o coeficiente de correlación proporciona la relación lineal que existe entre dos variables cualesquiera, como instrumento estadístico mide el grado de asociación lineal el

cual puede tomar un valor que comprendido entre el -1 a 1 (Lahura, 2003).

- Coeficiente de correlación > 0, entonces existe una correlación positiva, entonces las variables covarían linealmente y positivamente.
- Coeficiente de correlación ≈ 0, no existe ningun tipo de correlación, no varían linealmente.
- Coeficiente de correlación < 0, entonces existe una correlación negativa, entonces las variables covarían linealmente y negativamente.

4.2.3.1 Matriz de correlación.

Una matriz de correlación es una tabla que muestra los coeficientes de correlación entre varias variables, está herramienta es útil para identificar y visualizar patrones de datos. En la **Figura 5** se muestra un ejemplo de matriz de correlación.

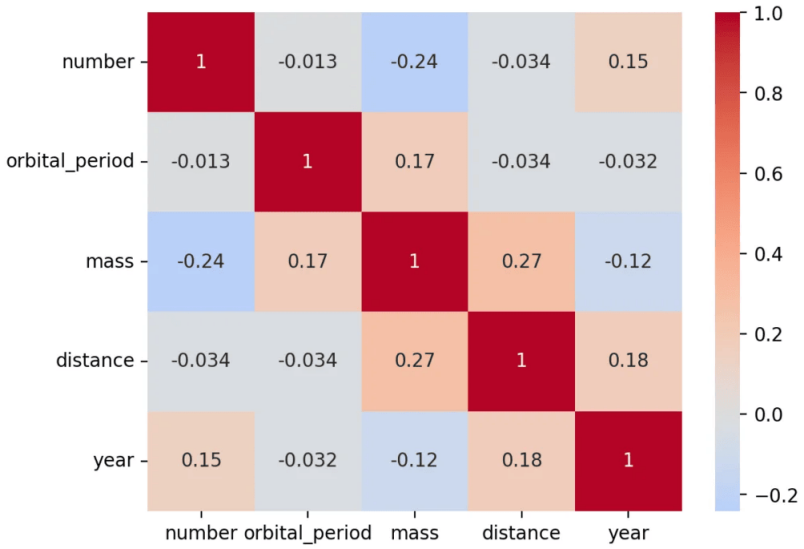


Figura 5. Ejemplo matriz de correlación.

Fuente: Rodríguez (2023).

4.2.3.2 Factor de inflación de la varianza (VIF).

Según Mandeville (2008) el factor de inflación de varianza, es una herramienta sencilla y directa para diagnosticar la presencia de multicolinealidad y cuantificar el impacto causado por la falta de independencia, mediante la siguiente ecuación:

$$VIF_i = \frac{1}{1 - R_i^2} \tag{1}$$

Donde:

R_i : Coeficiente de determinación.

Se define multicolinealidad como la relación lineal entre dos o mas variables independientes de un modelo (Gujarati & Porter, 2009).

El valor del VIF es siempre positivo y crece en un conjunto de datos a medida que aumenta la multicolinealidad entre las características. En la **Tabla 2** se muestra la interpretación según el valor del VIF.

Tabla 2. Interpretación del valor VIF dependiendo la multicolinealidad.

Valor VIF	Grado de multicolinealidad
Hasta 5	Débil/Moderado
De 5 a 10	Elevado
Mayor a 10	Muy elevado

Fuente: Gil (2020).

4.2.4 Data Sampling

En el procesado de los datos previo al entrenamiento, habitualmente existe desequilibrio de distribución de datos en las diferentes etiquetas, por lo que se usa los métodos de resampling los cuales están diseñados para agregar o eliminar ejemplos de un conjunto de datos de entrenamiento y lograr un equilibrio en la distribución de los datos, existen dos posibles casos de resampling de datos sobremuestreo (Oversampling) y submuestreo (Undersampling) (Brownlee, 2020).

4.2.4.1 Sobremuestreo.

Este método consiste en crear o duplicar los datos de la clase minoritaria hasta alcanzar un equilibrio con la clase mayoritaria (Torres et al., 2021).

4.2.4.2 Submuestreo.

El método de submuestreo a diferencia del anterior consiste en eliminar datos de la clase mayoritaria para lograr un equilibrio con la clase minoritaria.

El método de sobremuestreo es el más usado debido que agrega datos a la clase minoritaria evitando así perder datos muy valioso como se da con el submuestreo. En la **Figura 6** se observa como trabajan los métodos de remuestreo antes mencionados.

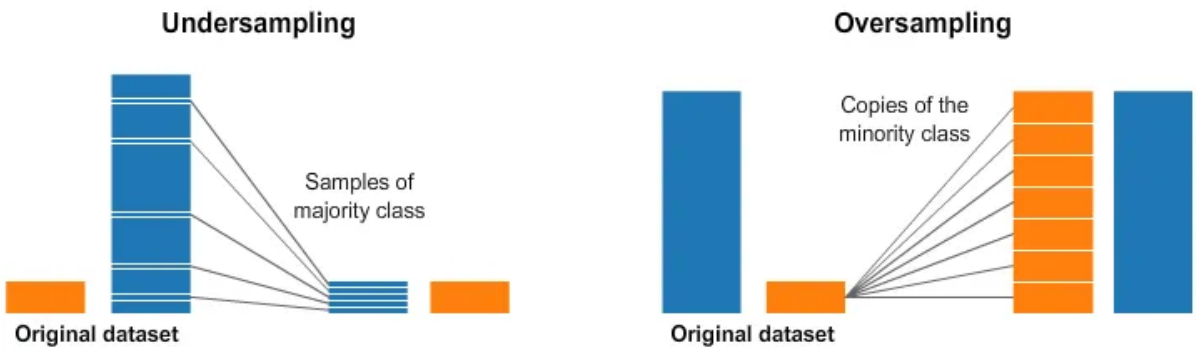


Figura 6. Balanceo de Datos, Undersampling y Oversampling.

Fuente: Al-Rahman (2021).

4.2.4.3 Synthetic Minority Oversampling Technique (SMOTE)

SMOTE primero selecciona una muestra de clase minoritaria a al azar y encuentra sus k vecinos de clase minoritaria más cercanos. Luego, la instancia sintética se crea eligiendo uno de los k vecinos más cercanos b al azar y conectando a y b para formar un segmento de línea en el espacio de características (Hoens & Chawla, 2013).

Las instancias sintéticas se generan como una combinación convexa de las dos instancias elegidas a y b.

4.2.5 Normalización

Debido al gran cálculo computacional que requiere entrenar modelos de IA, se debe comprimir o extender los datos a rangos definidos, por lo que se puede hacer uso de diferentes tipos de normalización de datos.

En este contexto, supongamos que tenemos un base de datos $X[j, i]$ donde $X[:, i]$ representa las características o variables (columnas), mientras $X[j, :]$ representa las entradas o datos (filas), entonces se define los diferentes tipos de normalizado y como se aplican a la base de datos $X[j, i]$.

4.2.5.1 Normalización z o estandarización.

La normalización z, transforma los valores de una variable en un conjunto de datos de tal forma que tengan una media de cero y una desviación estándar de uno (Blanco, 2023), y se determina mediante la **ecuación 2**.

$$\hat{X}[:, i] = \frac{X[:, i] - \mu_i}{\sigma_i} \tag{2}$$

donde:

μ_i : Media aritmética.

σ_i : Desviación estándar.

La media aritmética o promedio μ es la suma de numérica de dos o más valores divido para el número de muestras N.

$$\mu_i = \frac{1}{N}\sum_{k=1}^{N} X[k,i]. \tag{3}$$

La desviación estándar σ es un índice de dispersión estadístico, que representa la variabilidad de una población de datos, si el valor de la desviación es bajo indica que los valores están cerca de la media, mientras si es elevado indica que los valores se distribuyen en rangos muy amplios.

$$\sigma_i = \sqrt{\frac{1}{N-1}\sum_{k=1}^{N}(X[k,i]-\mu_i)^2} \tag{4}$$

4.2.5.2 Normalización min-máx.

Blanco (2023) define como una técnica utilizada en estadísticas y aprendizaje automático para transformar las variables de un conjunto de datos en un rango específico, comúnmente entre 0 y 1, y se calcula mendiante la **ecuación 5**.

$$\hat{X}[:,i] = \frac{X[:,i] - min(X[:,i])}{max(X[:,i]) - min(X[:,i])} \tag{5}$$

4.3 Capítulo 3

En el presente apartado se considera proporcionar información sobre redes neuronales artificiales, y las diferentes formas de evaluar el aprendizaje de un modelo mediante métricas.

4.3.1 Deep Learning (DL)

El Deep Learning (aprendizaje profundo) es un subcampo del machine learning (aprenidzaje automático) que ocupa algoritmos que tratan de similar al cerebro humano para resolver una amplia categoría de IA moderna, se puede encontrar muchos ejemplos en teléfonos inteligentes como, detección de rostros, corrección automática, texto predictivo entre otros. En la **Figura 7** se observa un diagrama de la estructura de una IA.

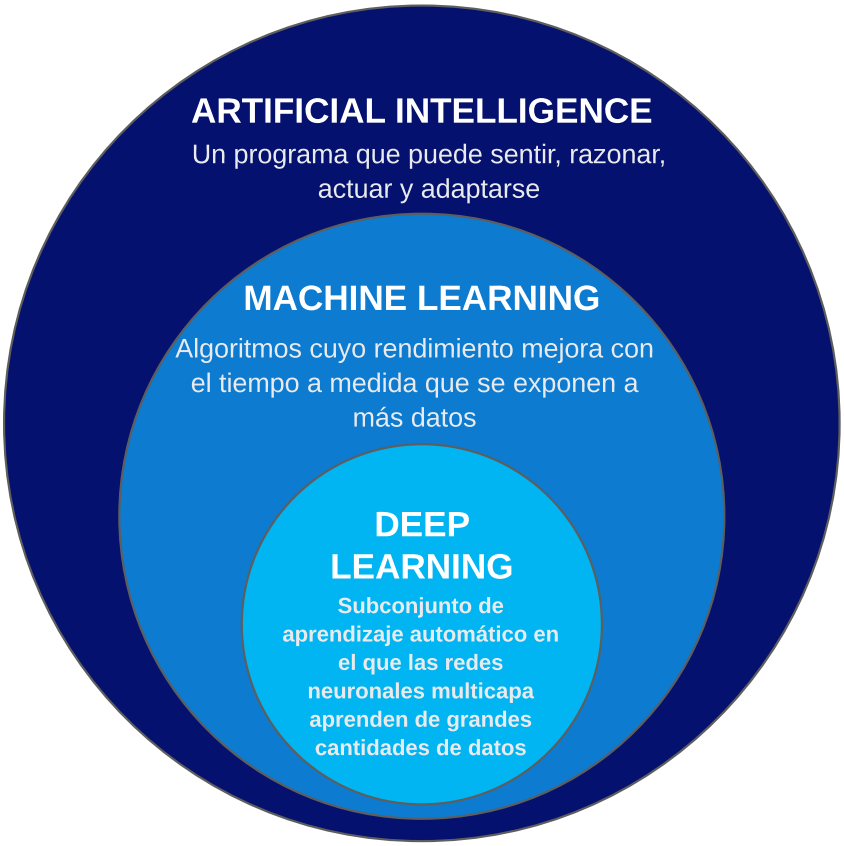


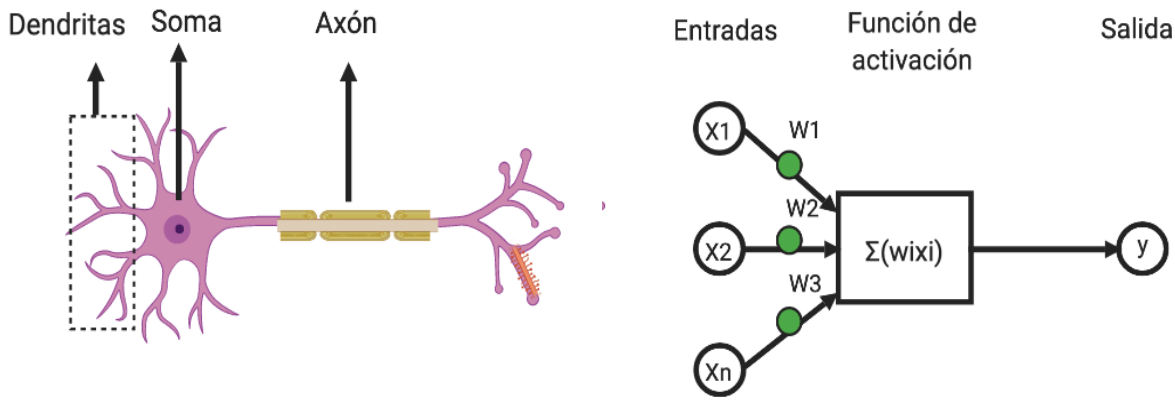
Figura 7. Clasificación de la Inteligencia Artificial.

Fuente: Alonso (2020).

El campo DL tiene su enfoque en el aprendizaje de representaciones apropiadas de datos. Como su termino "profundo" se refiere a la idea de aprender la jerarquía de conceptos directamente a partir de datos sin procesar, un termino más técnico seria ingeniería de características automatizadas (Ketkar & Moolayil, 2021).

4.3.1.1 Red Neuronal Artificial (ANN).

Una ANN es un modelo matemático que trata de imitar al cerebro humano mediante un conjunto de algoritmos. El concepto fue propuesto por primera vez por Warren McCulloch y Walter Pitts y luego implementado por Frank Rosenblatt como perceptrón, que es un modelo de una solo neurona igual a una neurona biológica (Politi, 2022).



(a) (b)

Figura 8. (a) Neurona, (b) Perceptrón.

Fuente: García (2019).

El perceptrón como ya se mencionó imita una neurona; en la **Figura 8** se visualiza está relación de entradas de datos, activación de neurona y salida en respuesta a los datos procesados.

Si observamos la parte del percetrón, $x_1, x_2, .., x_n$ representan la entrada de datos los cuales se multiplican por los pesos relativos w_i y este resultado se suma con el valor de sesgo (bios) obteniendo la función de entrada neta, y finalmente la función de activación f determina el cálculo de salida y si la neurona debe activarse o no, entonces se puede generalizar este proceso mediante la siguiente ecuación.

$$y = f(\sum_{i=1}^{n} x_i \cdot w_i + b) \tag{6}$$

Los pesos w, representa las conexiones entre un nodo y otro. El peso w_k es un número positivo si un nodo estimula a otro, o negativo si un nodo suprime a otro, los nodos con valores de pesos más altos tienen mayor influencia en los demás nodos.

4.3.1.2 Multilayer Perceptron (MLP).

En la **Figura 9**, se representa un multilayer percetron, de acuerdo Abirami y Chitra (2020) el MPL se compone de multiples neuronas conectadas entre sí mediante nodos a través de enlaces llamados sinapsis. Todo MLP consta de tres capas, la primera capa llamada capa de entrada, seguida por la capa oculta, la cual puede contener una o muchas capas y finalmente la capa de salida. Las neuronas de cada capa están interconectadas y asignadas un peso w_i el cual se va ir actualizando durante la fase de entrenamiento, en donde se comparan los valores de salida de la última capa con las etiquetas asignadas y la diferencia entre estos se conoce como

función de pérdida.

El proceso de aprendizaje se encarga de minimizar esta función mediante el algoritmo de back-propagation o retro-propagación el cual actualiza los pesos de cada conexión.

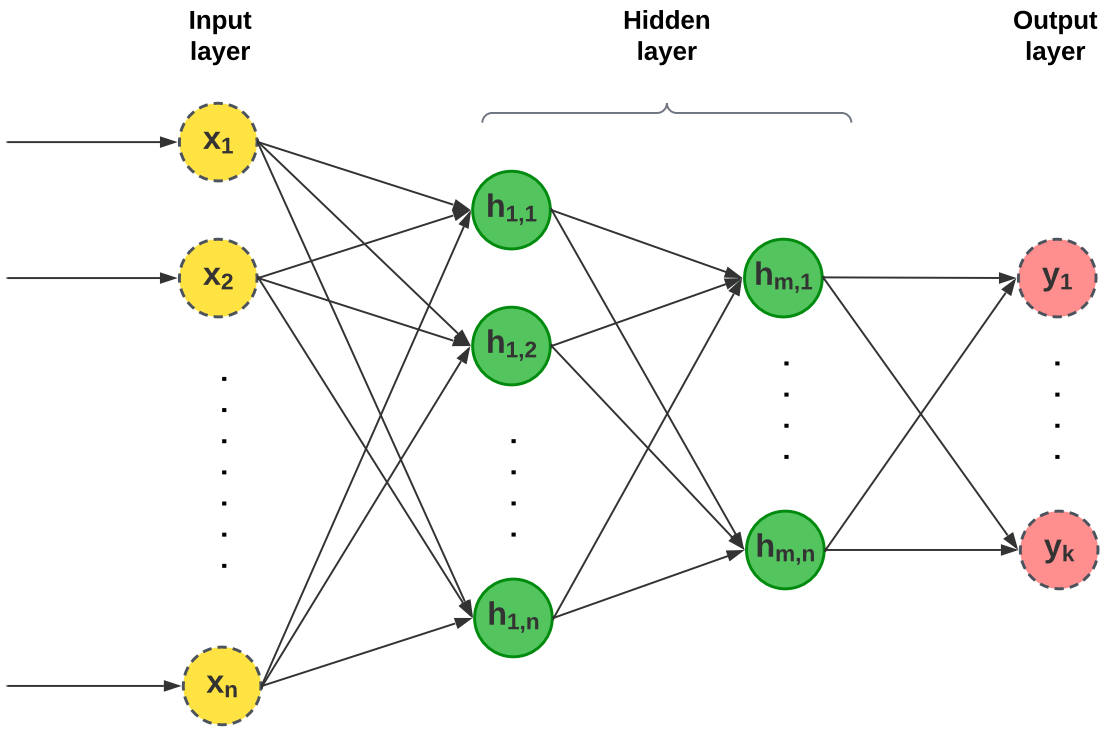


Figura 9. Arquitectura de una MLP.

4.3.1.3 Recurrent Neural Network (RNN).

La RNN es un tipo de red neuronal que usa datos secuenciales o series temporales, este tipo de red se caracteriza por su "memoria", dado que utilizan información de entradas anteriores para utilizar en las nuevas entradas y los resultados son independientes entre sí, los resultados de las RNN depende de los elementos anteriores dentro de la secuencia. Esté tipo de algoritmo es usado comúnmente en el procesamiento de lenguaje natural (npl), reconocimiento de vos, predicción de texto, entre otros (Ketkar & Moolayil, 2021). En la **Figura 10** se observa la estructura de una RNN.

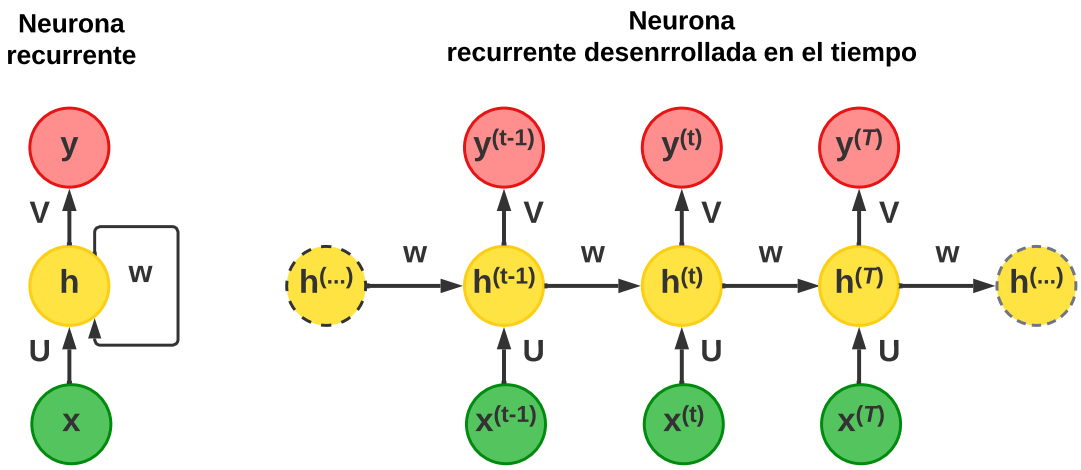


Figura 10. Estructura red neuronal recurrente.

Una RNN se puede describir mediante las **ecuaciones 7 y 8**:

$$h^{(t)} = \tanh(Ux^{(t)} + Wh^{(t-1)} + b) \tag{7}$$

$$\hat{y} = softmax(Vh^{(t)} + c) \tag{8}$$

donde:

x : Datos de entrada.

U : Pesos de activación en la entrada.

w : Pesos del estado oculto.

h : Estado oculto.

V : Pesos a la salida de la función de activación.

$\hat{y}$: Predicción.

4.3.1.3.1 Tipos de RNN.

Ketkar y Moolayil (2021) clasifica una RNN según la longitud de los datos de entrada y salida como se muestra en la **Figura 11**.

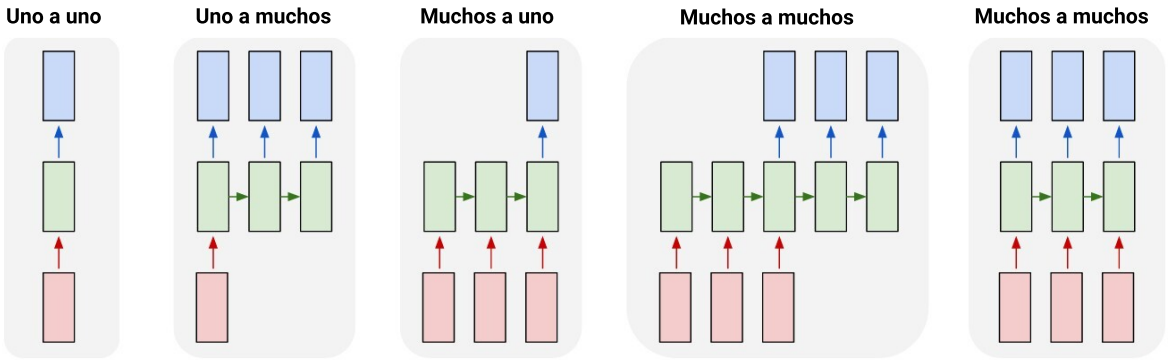


Figura 11. Tipos de RNN con base en su longitud de entrada y salida.

Fuente: Karpathy (2015).

4.3.1.4 Long Short Term Memory (LSTM).

Las LSTM fue propuesta por Hochreiter y Schmidhuber (1997) para tratar el problema de desaparición del gradiente que se produce en una RNN. Este tipo de red es capaz de predecir secuencias debido a su capacidad de aprender dependencias a largo plazo a partir de datos secuenciales, las conexiones de nodos se muestran en la **Figura 12**.

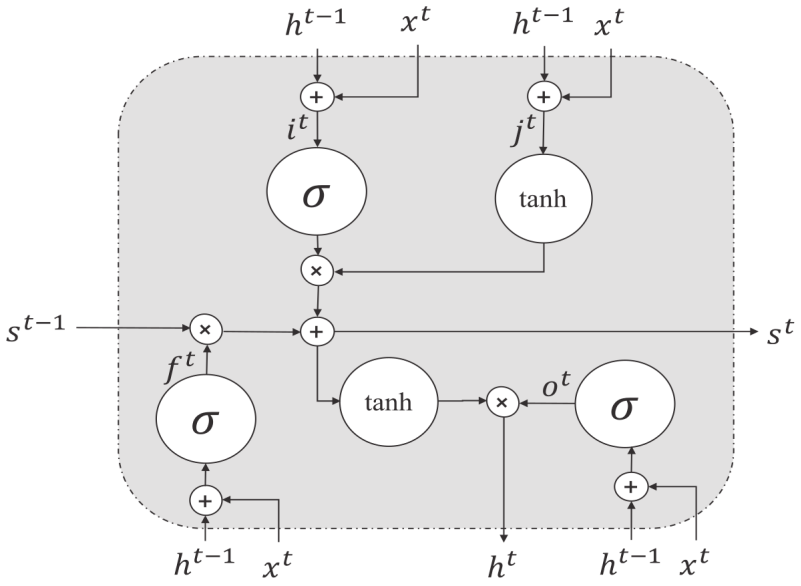


Figura 12. Diagrama de una unidad LSTM.

Fuente: Chen et al. (2021).

Sean i la puerta de entrada, j variable de control de la puerta de entrada, o puerta de salida, f puerta de olvido y s la celda de estado; $W = [W^i, W^j, W^f, W^f, W^o]$ representan los pesos de la matriz en la capa de entrada y $U = [U^i, U^j, U^f, U^o]$ los pesos de la capa oculta, $b = [b^i, b^j, b^f, b^o]$ representa el término de sesgo de los diferentes tipos de nodos, y $W \in \mathbb{R}^{m \times n}$,

$U \in \mathbb{R}^{m \times m}$ y $b \in \mathbb{R}^m$ donde m es la longitud de la secuencia de entrada y n es el número de capas ocultas.

El cálculo de la red LSTM en un momento t se define mediante las **ecuaciones 8-14**.

$$i^t = \sigma(W^i x^t + U^i h^{t-1} + b^i) \tag{9}$$

$$j^t = g(W^j x^t + U^j h^{t-1} + b^j) \tag{10}$$

$$f^t = \sigma(W^f x^t + U^f h^{t-1} + b^f) \tag{11}$$

$$s^t = f^t \otimes s^{t-1} + i^t \otimes j^t \tag{12}$$

$$o^t = \sigma(W^o x^t + U^o h^{t-1} + b^o) \tag{13}$$

$$h^t = o^t \otimes g(s^t) \tag{14}$$

Donde:

σ : Es la función de activación, predeterminado función $sigmoide$.

g : Función de activación, predeterminado función $\tanh$.

$\otimes$: Producto Hadamard (multiplicación de vectores a nivel de elementos).

4.3.1.5 Gated-Recurrent-Units (GRU).

Según Li et al. (2021) las unidades GRU son similares a las LSTM en que usan puertas pero tienen menos parámetros como se muestra en la **Figura 13**, y mantienen la información mas relevante para la predicción.

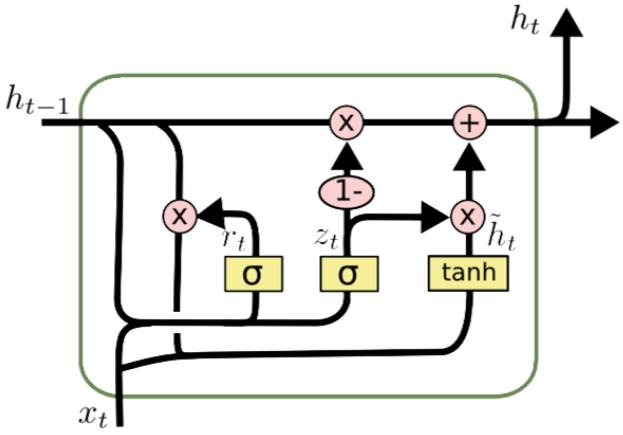


Figura 13. Diagrama de una unidad GRU.

Fuente: Li et al. (2021).

El cálculo de la red GRU se define mediante las **ecuaciones 14-18**. Sea r puerta de

reinicio, z puerta de actualización y h estado candidato entonces:

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z) \tag{15}$$

$$r_t = g(W_r x_t + U_r h_{t-1} + b_r) \tag{16}$$

$$\tilde{h_t} = tanh(W_h x_t + U_r(r_t \otimes h_{t-1})) \tag{17}$$

$$h_t = (1 - z_t) \otimes h_{t-1} + z_t \otimes \tilde{h_t} \tag{18}$$

4.3.1.6 Transfer learning (TL)

Byrnes (2007) define al TL como la capacidad de ampliar lo aprendido de un contexto a nuevos contextos.

Para definir el TL Pan y Yang (2010) introduce las definiciones de:

- Dominio: Un dominio D es un tupla $(\chi, P(X))$. Donde χ es el espacio de características de D y $P(X)$ es la distribución marginal para $X = \{x_1, ..., x_n\} \in \chi$.
- Tarea: Una tarea T es una tupla $(Y, f())$ para un domindo dado D; Y es el espacio de etiquetas de D y $f()$ es una función predictiva objetiva para D; $f()$ no se proporciona, pero se aprende de los datos de entrenamiento $f() = P(y|x)$.

Entonces el TL se define como: dado un conjunto de dominios de origen $DS = D_{S_1}, ..., D_{s_n}$ donde $n > 0$, un dominio objetivo D_t, un conjunto de tareas origen $TS = T_{S_1}, ...T_{S_n}$ donde $T_{S_i} \in TS$ y $D_{s_i} \in DS$, y una tarea objetivo T_t que corresponde a D_t, el TL ayuda a mejorar el aprendizaje de la función predictiva objetivo $f_t()$ en D_t donde $D_t \notin DS$ y $T_t \notin TS$.

4.3.2 Overfitting y Underfitting

Al momento de evaluar un modelo de IA, se debe tener en cuenta que el modelo no esté sobreajustado o subajustado: Gonzalez (2023) los define overfitting y underfitting como:

4.3.2.1 Overfitting o Sobreajuste.

Se refiera a que un modelo se encuentra muy ajustado a los datos de entrenamiento incluyendo el ruido de dichos datos, causando que dicho modelo no sea capaz de reconocer patrones generales que sean extrapolables a nuevos datos.

4.3.2.2 Underfitting o Subajuste.

El subajuste se refiera a que el modelo es incapaz de reconocer patrones en los datos de prueba, debido a un modelo muy sencillo.

En la **Figura 14 s**e observa los diferentes tipos de ajuste que puede tener un modelo y como este es capaz de interpretar los patrones.

Figura 14. Sobreajuste y subajuste de un modelo.

Fuente: Gonzalez (2023).

4.3.3 Métricas de Evaluación

Una métrica de evaluación cuantifica el rendimiento del modelo predictivo. Para los problemas de predicción las métricas comparan la etiqueta de la clase con la etiqueta predicha por le modelo. Las medidas de evaluación desempeñan un papel muy importante en la evaluación del rendimiento del modelo.

Existen una gran variedad de métricas estándares como accuracy, recall entre otras; la mayoría de métricas se pueden comprender de mejor manera utilizando una clasificación binaria, entonces se tiene dos clases 1 y 0 o clase positiva y clase negativa respectivamente. Haciendo uso de una matriz de confusión sobre el rendimiento de un modelo se tiene:

Tabla 3. Matriz de confusión binaria.

	Predicción Positiva	**Predicción Negativa**
Clase Positiva	VP	FN
Clase Negativa	FP	VN

Donde:

- VP (Verdaderos Positivos): La clase predicha es 1 y la real es 1.
- FN (Falso Negativo): La clase predicha es 0 y la real es 1.
- FP (Falso Positivo): La clase predicha es 1 y la real es 0.
- VN (Verdadero Negativo): La clase predicha es 0 y la real es 0.

4.3.3.1 Accuracy o Exactitud.

Es una de las métricas más utilizadas, y mide el porcentaje cuando una clase pronosticada no coincide con la clase esperada, se calcula mediante la **ecuación 19**.

$$Accuracy = \frac{Predicciones\ Correctas}{Total\ de\ Predicciones} = \frac{VP + VN}{VP + VN + FN + FP} \tag{19}$$

4.3.3.2 Recall o sensibilidad.

Evalúa que tan bien se predijo la clase positiva. Ver **ecuación 20**.

$$Recall = \frac{VP}{VP + FN} \tag{20}$$

4.3.3.3 Specifity o Especificidad.

Es lo contrario al Recall, evalúa que tan bien se predijo la clase negativa. Ver **ecuación 21**.

$$Specifity = \frac{VN}{VN + FP} \tag{21}$$

4.3.3.4 Precision o Precisión.

Es la relación entre las predicciones clasificadas como positivas y el número total de la clase positiva. Ver **ecuación 22**.

$$Precision = \frac{VP}{VP + FP} \tag{22}$$

4.3.3.5 F1-score.

Proporciona una manera de combinar las métricas *precision* y *recall* en una solo medida que captura ambas propiedades, dado que, por sí solos, no muestran el comportamiento completa.

Según Brownlee (2020), podemos tener una excelente *precision* con un *recall* terrible, o biseversa. El *F1-score* proporciona una manera de expresar ambas inquietudes con una sola puntuación, por lo que es óptima para aplicar a datos desbalanceados. La medida *F1-score* se calcula mediante la **ecuación 23**:

$$F1 - score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{23}$$

4.3.3.6 Curva ROC.

La curva ROC(Curva Caracteristica Operativa Receptor) es un gráfico que resume el rendimiento de los modelos sobre la clase positiva, el eje X indica la tasa de falsos positivos y el eje y indica la tasa positivos verdaderos, la gráfica se define en el intervalo de 0 a 1, en la **Figura 15** se logra observar como trabaja la curva ROC.

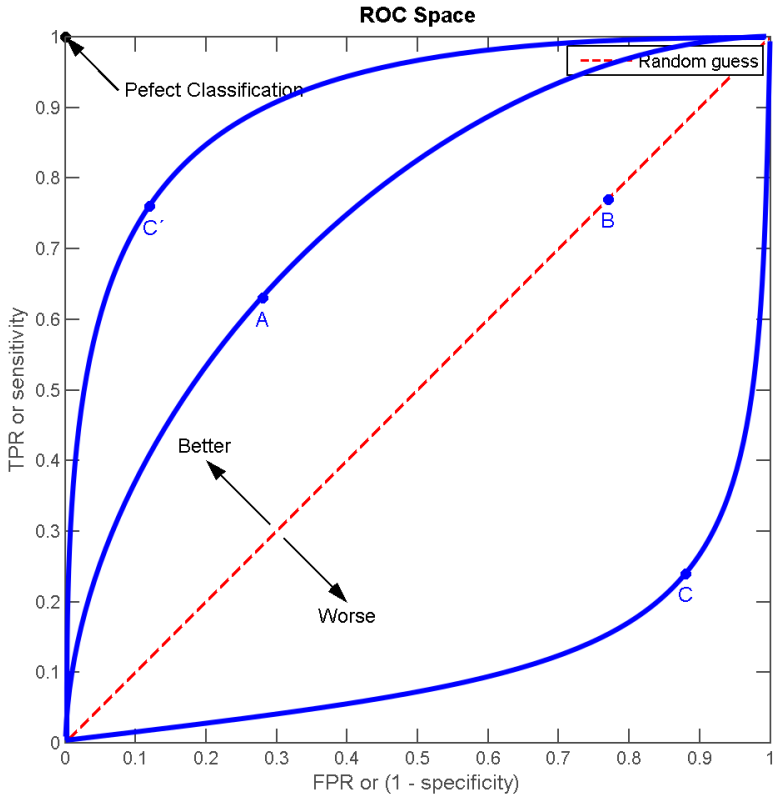


Figura 15. Curva ROC.

Fuente: E. Melillanca (2018).

4.4 Capítulo 4

Finalmente, el cuarto capítulo aborda los conceptos principales de una interfaz gráfica y su uso para la gestión de información.

4.4.1 Interfaz gráfica de usuario GUI

La interfaz gráfica de usuario (GUI) es un tipo de interfaz de usuario que utiliza un conjunto de imágenes y objetos gráficos para representar la información; está conformado por dos componentes: entrada y salida. La entrada se encarga de recibir las necesidades de una persona, y la salida es la forma en que el computador transmite los resultados procesados solicitados

por el usuario (Albornoz, 2014). En la **Figura 16**, se puede visualizar el papel que cumple la interfaz gráfica en la comunicación usuario-computador.

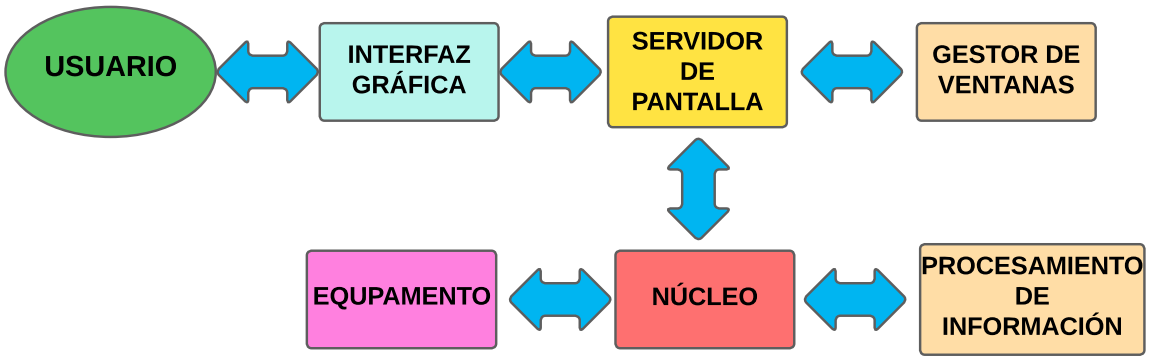


Figura 16. Capas de interfaz gráfica de usuario.

4.4.1.1 Dashboard.

Un dashboard es una herramienta visual de datos para gestionar la información, el análisis y la monitorización de métricas para hacer un seguimiento del estado de una empresa, un departamento, una campaña o procesos en específicos (Ortiz, 2022). En la **Figura 17** se muestra un ejemplo de dashboard.

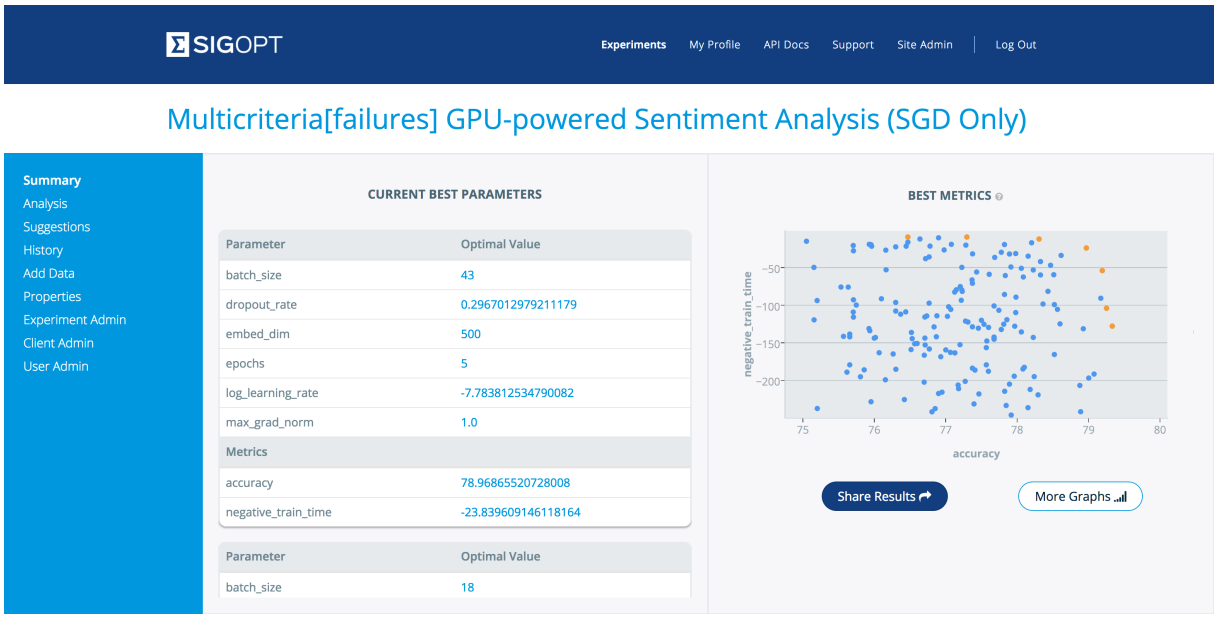


Figura 17. Ejemplo dashboard.

Fuente: Tartakovsky et al. (2017).

5. Metodología

5.1 Área de trabajo

5.1.1 Localización

El CEV objeto de estudio, se encuentra ubicado en la provincia de Loja, cantón Loja en las coordenadas UTM 693030 E 9558392 N y 693526 E 9556476 N, vía antigua Loja-Catamayo. Su ubicación geográfica y altitud sobre el nivel del mar se muestran en la **Figura 18**.

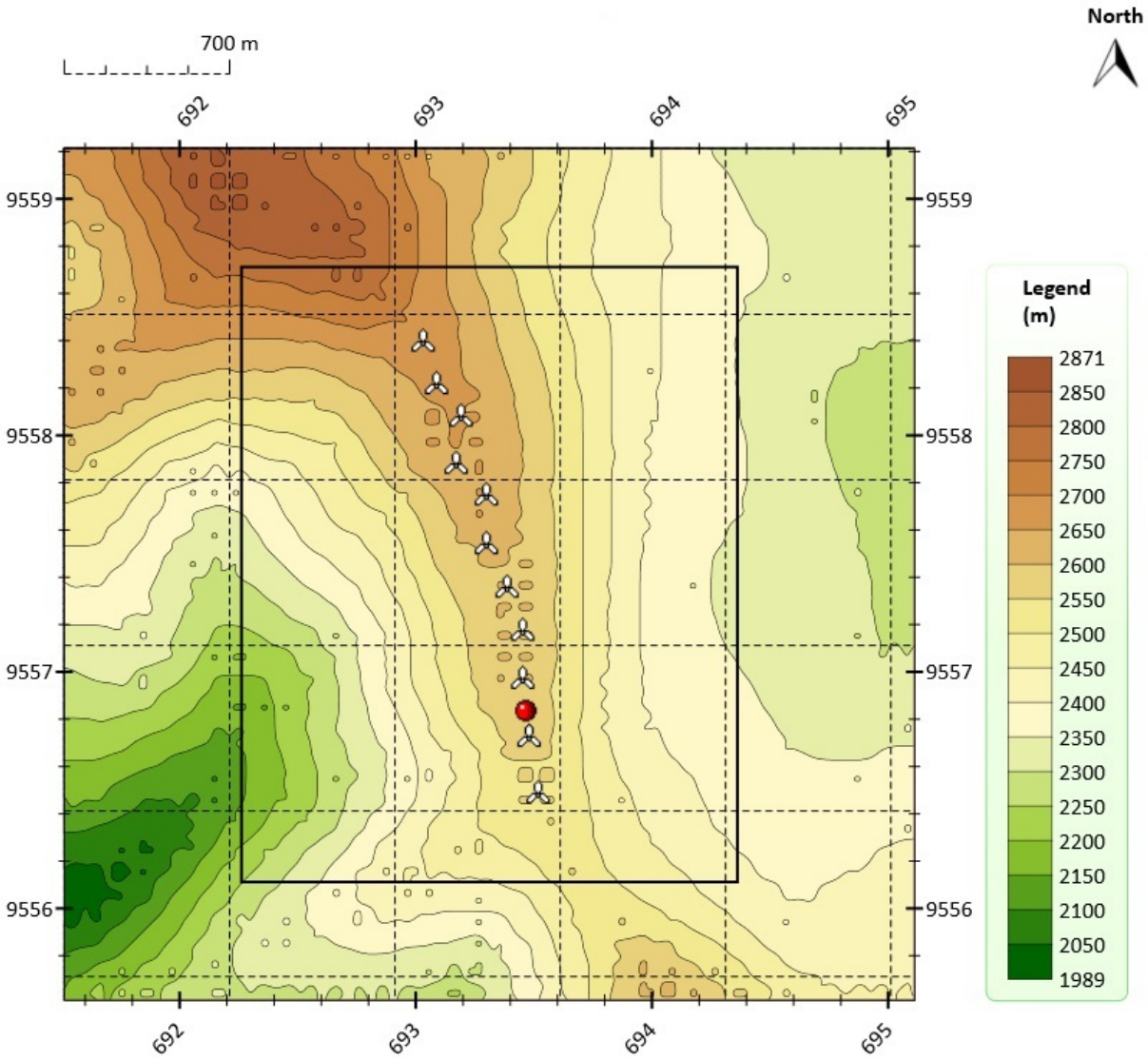


Figura 18. Ubicación CEV.

Fuente: Ayala et al. (2017).

5.2 Equipos y materiales

5.2.1 Equipos

El principal hardware con el que se cuenta es una computadora cuya características son:

- Marca: ASUS.
- Modelo: F15.
- CPU: 12th Gen Intel i7-12700H (20).
- GPU: NVIDIA GeForce RTX 3050 Mobile.
- Memoria ram: 16GB.
- Capacidad de almacenamiento: 1.0TB.
- Sistema operativo: GNU/linux Ubuntu 22.04.3 LTS x86_64.

5.2.2 Materiales

5.2.2.1 Base de Datos.

Los datos de la CEV recogidos por el sistema SCADA en formato .txt correspondiente al periodo comprendido entre los años 2014 y 2021, recogidos en un intervalo de 10 min, adicionalemente se cuenta con el registro de alarmas en el cual se encuentra la fecha, hora, duración y tipo de paro que el sistema SCADA registro durante una alarma.

5.2.2.2 Python.

De acuerdo al sitio oficial de python define “Python es un lenguaje de programación potente y fácil de aprender. Tiene estructuras de datos de alto nivel eficientes y un simple pero efectivo sistema de programación orientado a objetos. La elegante sintaxis de Python y su tipado dinámico, junto a su naturaleza interpretada lo convierten en un lenguaje ideal para scripting y desarrollo rápido de aplicaciones en muchas áreas, para la mayoría de plataformas.”(Python Software Fundation, 2023)

El lenguaje python contiene una amplia biblioteca de análisis de datos, inteligencia artificial y desarrollo de interfaces gráficas para el presente trabajo de titulación se uso:

- Pandas.
- Numpy.
- Matplotlib.
- Scikit-learn.
- Imbalanced-learn.
- Flask.

- Dash.
- Plotly.
- Pytorch.

5.2.2.3 Latex.

El sitio web oficial Proyecto LaTex (2023) define a LaTeX como un paquete de macros que permite al autor la composición tipográfica de alta calidad, diseñado para la producción de documentación técnica y científica, es muy utilizado en la publicación de documentos científicos. Adicionalmente es un software libre y se encuentra disponible para Linux, Windows, Mac Os y Online.

5.3 Procedimiento

El procedimiento metodológico que se siguió en este trabajo de titulación se muestra en el flujograma de la **Figura 19**.

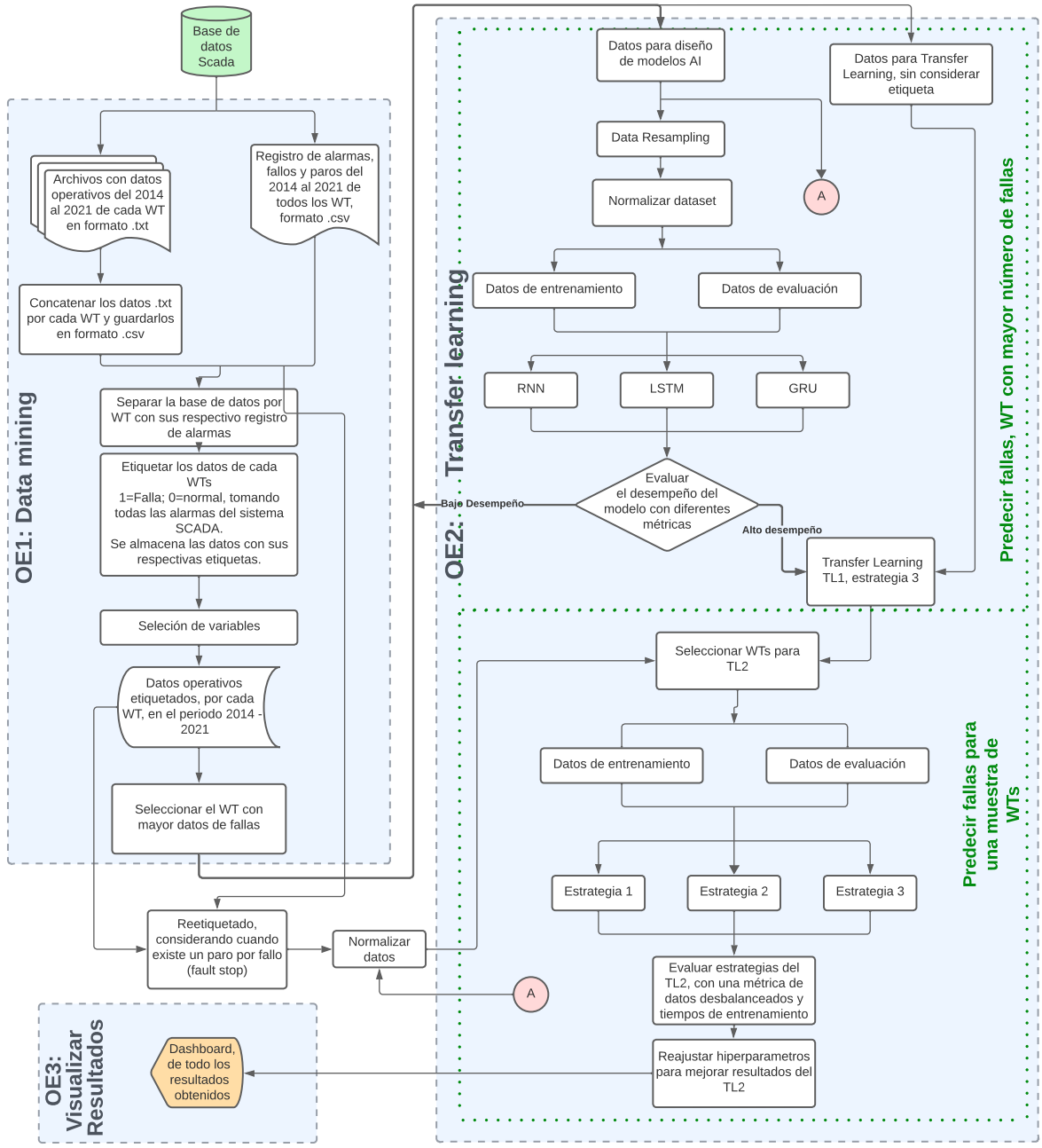


Figura 19. Flujograma para el desarrollo del trabajo de titulación.

A continuación se describe el proceso metodológico empleado en cada objetivo específico.

5.3.1 OE1. Implementar Data Mining al registro histórico del periodo 2014-2021 de los datos SCADA y el registro de alarmas de fallos del parque eólico Villonaco del cantón Loja

Como se observa en la **Figura 19**, para el desarrollo del primer objetivo se utilizó un enfoque cuantitativo, debido a que se identificaron las variables y valores registrados por el

sistema SCADA de la CEV.

Los datos utilizados en este trabajo de integración curricular fueron proporcionados por la Empresa Pública Corporación Eléctrica del Ecuador (CELEC EP) bajo un acuerdo de confidencialidad, esta información se encuentra estructura de la siguiente manera (**Figura 20**):

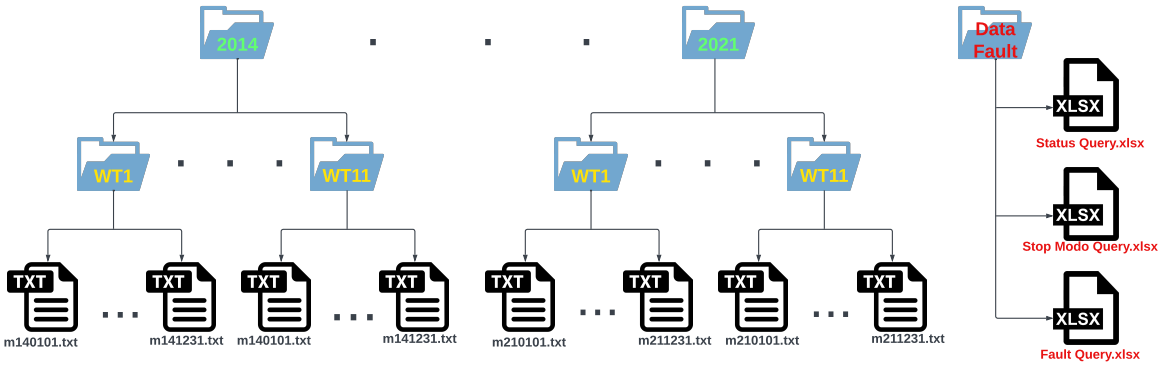


Figura 20. Estructura de los datos entregados por CELEC EP.

Contabilizando los archivos con la información del sistema SCADA y registro de fallas se tiene:

- 29085 archivos en formato .txt con información diaria de las variables de operación del sistema SCADA de los WTs, comprendidos desde al año 2014 al 2021.
- 3 archivos .xlsx con el registro de fallos de los WTs, en el mismo periodo mencionado anteriormente.

Luego, se revisó la dimensionalidad con el propósito de conocer la cantidad de variables que contenía cada archivo. A continuación, se eliminaron los datos atípicos, outlayers y valores vacíos o huecos (NaN).

Finalmente, utilizando técnicas de Data Mining se concatenaron los archivos con el objetivo de contar con un dataset consolidado que permita desarrollar el OE2. Es necesario precisar que en este proceso no se aplicaron técnicas estadísticas de muestreo, debido a que se trabajó con todos los datos.

Una vez concatenada la base de datos, se seleccionaron las principales variables para reducir la dimensionalidad, y con ayuda del registro de mantenimiento se identificaron las fechas en las que existieron fallas en diferentes componentes del aerogenerador y se aumentó una columna con registros binarios (0 = normal, 1 = falla).

5.3.2 OE2. Aplicar transfer learning para predicción de fallas y evaluar su rendimiento mediante diferentes métricas

El segundo objetivo también tuvo un enfoque cuantitativo, la base de datos se obtiene del primer objetivo y se toma una muestra. Para la selección de la muestra se consideró:

- A partir de la base de datos se la separó por WT.
- De la columna de registros binarios (label) se contaron las etiquetas con valor = 1.
- Como se muestra en la **Figura 19** se toma al WT con mayor número de etiquetas = 1.
- Al WT con mayor número de etiquetas se guarda en una nueva base de datos "WTPreliminar.csv".

La nueva base de datos "WTPreliminar.csv" se divide en varias regiones A, B y C la selección de regiones se realiza de la siguiente manera:

- Se considera todos los datos como una serie temporal.
- La región C se esta compuesta por todos los registros (etiquetas 1) entre el inicio de la alarma o fallo hasta cuando termino la alarma, es decir todo el intervalo de duración de la alarma registrado por el sistema SCADA.
- Los datos con etiquetas = 0 se dividen en las regiones A y B.
- La región B esta conformado por todos los datos que se encuentran en un intervalo de tiempo igual antes que el sistema registre el inicio de la alarma.
- La región A esta conformado por el resto de datos.

En la **Figura 21** se muestra un intervalo de datos del sistema SCADA y como se dividen las regiones, en el cual la región A representas datos normales, la región B representa datos que se los retira temporalmente dado que a estos datos se le va a aplicar el primer TL, adicionalmente la región B se compone de intervalos de tiempo igual para toda la base de datos y la región C son los datos cuando fallo el WT, este proceso se detalla más a detalle en la sección procesamiento (5.4.2.1), por lo tanto, la base de datos para entrenar el modelo basa va a estar conformado por las regiones A y C, y así mismo a estas regiones se aplican algoritmos de remuestreo.

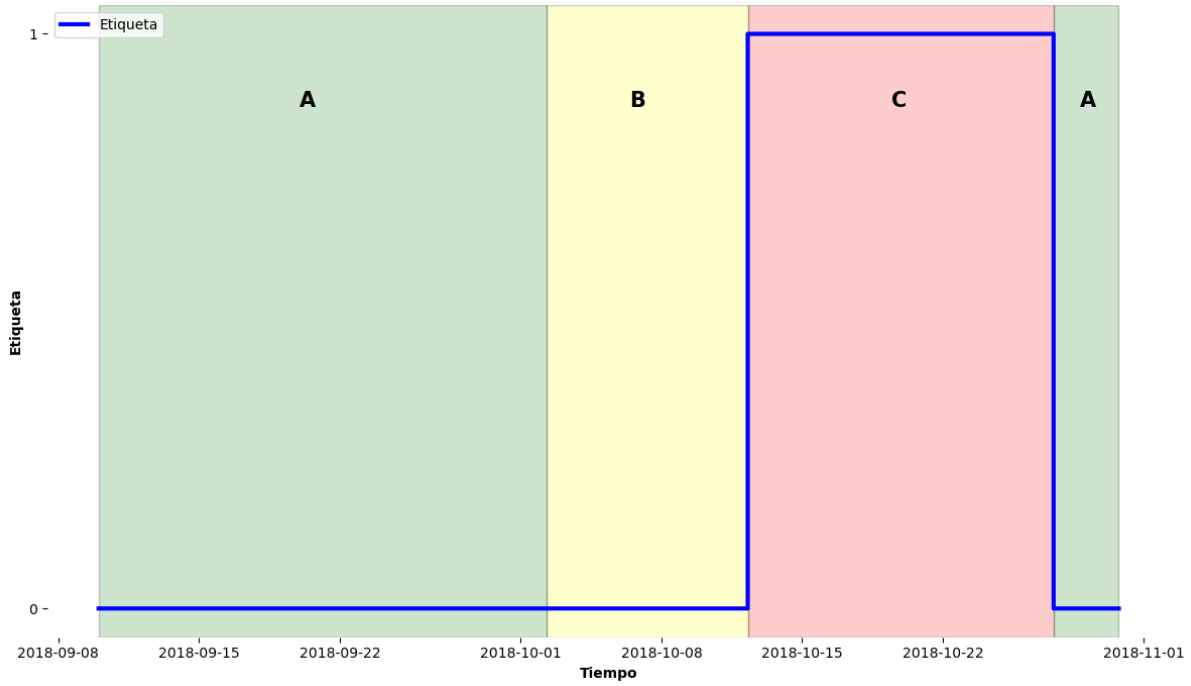


Figura 21. Fragmento de base de datos y muestra de selección de regiones A-B-C.

A partir de los datos remuestreados, se realizaron divisiones para obtener conjuntos de entrenamiento y evaluación. Se procedió a entrenar tres modelos de redes neuronales: RNN, LSTM y GRU, utilizando los datos de entrenamiento y posteriormente, se evaluaron estos modelos con el conjunto de datos de evaluación.

Inicialmente, se exploraron hiperparámetros de manera aleatoria, y a medida que avanzaba el proceso, se realizaron ajustes para mejorar el rendimiento y elevar los valores de las métricas en comparación con los resultados iniciales, si durante el proceso se observa que ninguno modelo presentaba buenos resultados se repetía el proceso de remuestreo aplicando diferentes algoritmos (seleccionado de color verde con líneas punteadas Predecir Fallas, WT con mayor número de fallas mostrados en la **Figura 19**).

Finalmente, se seleccionó el modelo que mostró los mejores resultados, y este modelo se utilizó durante todo el proceso de TL.

Para el TL1, se consideró una estrategia de TL y se evaluó los datos de la región B (Ver **Figura 21**), anticipando de esta manera un fallo mucho antes que el sistema SCADA lo registre.

En el Segundo Transfer Learning (TL2), se reutiliza el modelo base seleccionado en el TL1. A diferencia del TL1 se seleccionó una muestral aleatoria de los WTs, descartando el que fue usado para crear el modelo base; con dicha muestra se realiza un reetiquetado, pero en este caso solo se considera un tipo de fallo, y a la nueva base de datos no se aplican ningún algoritmo

de resmuestrreo.

Con la base de datos para el TL2 se divide en datos de entrenamiento y de evaluación, con los datos de entrenamiento se aplica TL con ayuda del modelo base y las diferentes estrategias de TL, a partir de modelo reentrenado se evalúa con los datos de evaluación y con una métrica para datos desbalanceados, se analiza los tiempos de entrenamiento y el valor de la métrica de evaluación; se reajusta los hiperparámetros para mejorar los resultados y del mejor resultado se aplica un TL a los datos del WT base.

A partir de estos procesos los resultados se presentan en el OE3.

5.3.3 OE3. Desarrollar un dashboard para el manejo y visualización de los resultados obtenidos

El último objetivo adopta un enfoque mixto. Desde la perspectiva cuantitativa, se destaca que no se aplican muestreos, ya que los datos manipulados son exclusivamente para fines de visualización. Estos datos comprenden los resultados obtenidos en todos los objetivos específicos anteriores; por otro lado, el enfoque cualitativo se basa en diversos análisis que caracterizan los modelos como buenos o malos.

Por lo que se programa un dashboard tratando de que sea amigable con el usuario, óptimo y sobre todo fácil de usar y capaz de mostrar el trabajo realizado; además, se busca mejorar la presentación de la información al usuario para una comprensión más efectiva.

5.4 Procesamiento y análisis de datos

Al igual que la sección anterior, el procesamiento y análisis de datos se desarrolló mediante el flujograma mostrado en la **Figura 19**.

5.4.1 OE1. Implementar Data Mining al registro histórico del periodo 2014-2021 de los datos SCADA y el registro de alarmas de fallos del parque eólico Villonaco del cantón Loja

5.4.1.1 Base de Datos.

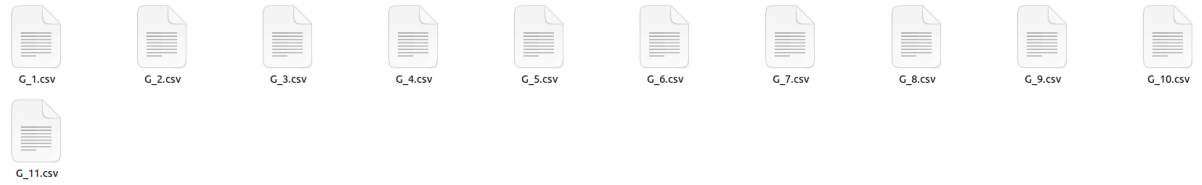
Los datos recogidos por el sistema SCADA se encuentran en formato txt, desde el 2014 hasta el 2021, sistema registra 64 variables hasta el 2017 y a partir de este año registra 68 variables y suman un total de más de 4 millones de datos. En la **Tabla 4** se observa una pequeña porción de los datos y algunas variables registradas por el sistema SCADA. Al usar Python leemos los archivos .txt para luego convertirlos en formato .csv y así posteriormente poder trabajar con dichos datos.

Tabla 4. Muestra de datos entregados por CELEC EP.

timestamp	wind_speed_avg	wind_speed_max	wind_speed_min	grid_active_power_avg	grid_active_power_max	grid_active_power_min
2014-01-01 00:00:00	4.690000	6.090000	2.610000	57.860000	103.290000	23.040000
2014-01-01 00:10:00	2.900000	4.760000	1.230000	7.630000	41.820000	-31.580000
2014-01-01 00:20:00	2.660000	4.430000	0.610000	4.010000	27.310000	-5.120000
2014-01-01 00:30:00	4.440000	6.040000	2.430000	50.140000	96.460000	-4.260000
2014-01-01 00:40:00	4.360000	5.690000	2.260000	51.060000	79.380000	-1.700000
2014-01-01 00:50:00	3.830000	5.160000	2.610000	30.520000	53.780000	17.070000
2014-01-01 01:00:00	3.920000	5.890000	2.680000	50.250000	86.210000	23.900000
2014-01-01 01:10:00	5.400000	7.160000	3.480000	107.390000	141.700000	82.800000
2014-01-01 01:20:00	5.770000	7.340000	3.860000	116.260000	156.210000	78.530000
2014-01-01 01:30:00	5.380000	6.760000	3.430000	106.390000	136.580000	61.460000

Como se mencionó, existe una diferencia de la cantidad de variables, por lo que para tener un dataset completo se tuvo que agregar valores NaN y lograr que todos los archivos tengan la misma dimensionalidad $(n \times 68)$, donde n son el total de registros, tomados durante todos los años cada 10 min.

Una vez obtenido la base de datos que abarca todos los WTs, se procede a segmentarlos en archivos individuales como se ilustra en la **Figura 22(a)**; además, se lleva a cabo el procesamiento de los registros por fecha, utilizando la variable *timestamp* que indica la fecha y hora de cada registro en el formato *año-mes-día hora:minutos:segundos*, esta proceso (**Figura 22(b)**) facilita un mejor control y gestión de los datos.



(a)

timestamp	wind_speed_avg	wind_speed_max	wind_speed_min	grid_active_power_avg	grid_active_power_max	grid_active_power_min	generator_capacitors_
2014-01-01 00:00:00	3.91	5.89	1.83	43.71	101.58	9.39	
2014-01-01 00:10:00	3.77	5.64	1.83	32.32	74.26	8.53	
2014-01-01 00:20:00	3.13	4.76	1.26	14.00	42.68	-29.87	
2014-01-01 00:30:00	4.13	6.56	1.48	49.90	115.24	-0.85	
2014-01-01 00:40:00	4.28	6.04	2.03	52.76	81.95	17.92	

(b)

Figura 22. Base de datos por WT. (a) Archivos de cada WT en formato .cvs, (b) Datos WT1 con fecha (timestamp) en orden ascendente.

El proceso de separar por WT los datos se lo realizo para evitar sobrecargar la memoria RAM del computador y optimizar los cálculos al momento del etiquetado y pre-procesado que se abordan en los temas siguientes.

5.4.1.2 Registro de alarmas.

Para la primera y segunda tarea de etiquetado, se cuenta con el registro de alarmas y los modos de paro del 2014 hasta el 2021 de todos los WT, en el cual como dato principal tiene la fecha y hora cuando inicia el fallo (FaultStartTime) de un WT, tiempo de duración y cuando finaliza la alarma (FaultEndTime). En la **Tabla 5** observamos una pequeña muestra del registro de alarmas.

Tabla 5. Muestra del registro de alarmas entregados por CELEC EP.

WTG	code	FaultStartTime	FaultEndTime	duration	stop_cat	FaultDescription
1	107	2015-04-20 15:29:42.060	2015-04-20 18:51:40.543	0 days 03:21:58.483000	PROFIBUS	(107)Error_profibus_20# station power supply
1	420	2015-05-16 12:39:16.330	2015-05-16 12:50:06.360	0 days 00:10:50.030000	CONVERTER	(420)Error_converter IGBT cooling 1# fan feedback
1	435	2015-05-28 04:57:14.733	2015-05-28 06:12:13.717	0 days 01:14:58.984000	CONVERTER	(435)Error_converter not ready
1	52	2015-05-28 13:22:26.077	2015-05-28 13:41:33.170	0 days 00:19:07.093000	GRID	(52)Error_grid_LVRT over time
1	75	2015-07-27 06:24:39.463	2015-07-27 06:37:08.587	0 days 00:12:29.124000	OTHERS	75
1	217	2015-08-04 10:13:25.933	2015-08-10 01:27:26.337	5 days 15:14:00.404000	PICH	(217)Error_pitch blade angle comparing
1	217	2015-08-10 02:31:24.247	2015-08-10 02:32:08.730	0 days 00:00:44.483000	PICH	(217)Error_pitch blade angle comparing
1	217	2015-08-12 03:00:41.323	2015-08-12 03:03:08.497	0 days 00:02:27.174000	PICH	(217)Error_pitch blade angle comparing
1	217	2015-08-12 03:19:23.997	2015-08-12 03:21:51.683	0 days 00:02:27.686000	PICH	(217)Error_pitch blade angle comparing
1	217	2015-08-12 03:29:37.447	2015-08-12 03:32:08.120	0 days 00:02:30.673000	PICH	(217)Error_pitch blade angle comparing
1	217	2015-08-12 03:34:42.510	2015-08-12 03:37:13.040	0 days 00:02:30.530000	PICH	(217)Error_pitch blade angle comparing
1	120	2015-08-21 17:03:38.697	2015-08-21 18:34:22.213	0 days 01:30:43.516000	PROFIBUS	(120)Error_profibus_41# station diagnostic

5.4.1.3 Etiquetado de datos.

El etiquetado de datos, es muy importante para el entrenamiento de algoritmos supervisados como se referencia en el flujograma (**Figura 19**). Los datos se etiquetan como **0** "funcionamiento normal" y **1** en "fallo"; según Chen et al. (2021) se etiqueta los datos de acuerdo al registro de alarmas, donde el sistema SCADA arroga una alarma (FaultStartTime) a partir de ese instante se etiqueta como 1 hasta que vuelva que se recupere el estado normal (FaultEndTime), el resto de datos se etiquetan como 0. La **Figura 23** muestra el algoritmo que se utilizó.

Algoritmo 1: Etiquetado de datos (primer etiquetado)

Datos: Alarmas $\{A_{(I)}\}_{j=1}^{J}$, registro de la turbina $\{T_{(t)}\}_{j=1}^{J}$ donde J es el número de WTs, t la hora que registo el sistema SCADA e I es un intervalo de tiempo.

Resultados: Columna de etiquetas para cada registro $\{L_{(t)}\}_{j=1}^{J}$.

1 Todas las categorias de alarmas se consideran como un solo tipo de alarma.

2 Identificamos la fecha y hora que inicio t_i la alarma y la hora y fecha que se finalizo t_f la alarma.

3 **Para** cada registro de la WT T^j en el tiempo t **hacer**:

4 **Si** $A_{t_i}^j \leq T_t^j \leq A_{t_f}^j$ y $t_i \leq t \leq t_f$:

5 $L_t^j = \mathbf{1}$

6 **Caso contrario**:

7 $L_t^j = \mathbf{0}$

8 L_{t-1}^j se adjunta el nuevo registro L_t^j

9 Se repite todo el ciclo para cada instante t del sistema SCADA y para cada WT j

10 Finalmente se adjunta la matriz L al registro T como una columna llamada “label”.

Figura 23. Etiquetado de base de datos para determinar el WT con mayor número de fallas y alarmas.

Es importante mencionar que este etiquetado solo se utiliza para el entrenamiento del modelo base y el TL1, para el TL2 se explica en el acápite 5.4.2.6.2.

5.4.1.4 Manejo de datos faltantes.

La presencia de datos faltantes, representados como “NaN”, en columnas o filas puede plantear desafíos al entrenar un algoritmo, y es crucial abordarlos cuidadosamente. Según plantea Pineda (2021) en su libro, se propone tres enfoques para manejar los datos nulos, cuando los datos faltantes en una columna son insignificantes en comparación con el tamaño total de la base de datos, se pueden eliminar, sin embargo, si la cantidad de datos faltantes en una columna es demasiado alta, eliminar toda la columna puede ser una opción, aunque el autor advierte que esto podría resultar en la pérdida de información relevante.

Para superar este problema, se sugiere utilizar métodos de imputación de valores ausentes. *Scikit-learn* proporciona la clase *SimpleImputer*, que ofrece estrategias como la media, mediana, frecuencia y constante para abordar la imputación de valores faltantes.

5.4.1.5 Selección de variables a trabajar.

Conformada por 69 variables, la base de datos sugiere la conveniencia de reducir su tamaño, la razón principal radica en que trabajar con todas las variables entraña un costo computacional significativo; adicionalmente, existe cierta posibilidad de linealidad entre muchas de

estas variables. Dado que el objetivo es prever fallos en los WTs, se busca que los modelos tengan tiempos de respuesta rápidos, entonces un incremento en el costo computacional no solo impactaría la eficiencia, sino que también requeriría el uso de máquinas más costosas

En este sentido, se lleva a cabo la selección de variables que usa el método VIF con el uso de la **ecuación 1**, el cual mide la relación existente entre todas las variables. Para la selección de las variables se realiza un script en python, en donde se ingresan todo el conjunto de datos de las variables predictoras y predecida, y se evalúa el VIF conforme la **Tabla 2** lo indica, donde se seleccionan las variables con valores menores o iguales a 5 puntos, de esta manera se logra reducir las variables y evitar la multicolinealidad.

5.4.2 OE2. Aplicar transfer learning para predicción de fallas y evaluar su rendimiento mediante diferentes métricas

5.4.2.1 Divisón de regiones A,B y C

La división de regiones es crucial dado que de este proceso se va a obtener: la base de datos para crear el modelo base, el cual se conforma de las regiones A y C donde A son datos normales (etiqueta 0) y C datos de fallo o alarmas (etiquetas 1). B los datos para aplicar el TL1 (etiquetas 0), los que servirán para anticipar el fallo o alarma antes que el sistema SCADA lo registre asi mismo esta región nos permitira observar el comportamiento de una de las estrategias en datos similares al del conjunto de entrenamiento.

Para la división de las regiones se sigue el algoritmo mostrado en la **Figura 24**

Algoritmo 2: Selección de regiones

Datos: Registro de la WT $\{WTB_{(t)}\}$ y columna de etiquetas $\{LB_{(t)}\}$ del WT con mayor número de fallos (etiquetas = 1); donde t la hora y fecha que registo el sistema SCADA.

Resultados: Columna con etiquetas A, B y C para cada registro $\{E_{(t)}\}$.

Definir un intervalo de tiempo I_B para la región B.
Para cada tiempo t **hacer**:
    $t_i = t - (2 \cdot I_B)$.
    $[t_i, t)$
    Si $LB_t = 1$:
        $E_t =$ C.
    Caso contrario, Si $LB_t = 1$ y $LB_{[t_i,t)} \neq 1$:
        $E_{[t-I_B,t)} =$ B.
    Caso contrario:
        $E_t =$ A.
Finalmente se adjunta la matriz E al registro WTB como una columna llamada "Regiones".

Figura 24. División de regiones para el modelo base y TL1

5.4.2.2 Remuestreo.

Dado que las fallas son menos recurrentes en comparación con los datos de funcionamiento óptimo, surge un desbalance de datos. Para abordar este desbalance, se recurre a algoritmos de oversampling. Como se detalla en el flujograma (**Figura 19**), este proceso implica un bucle iterativo hasta alcanzar un algoritmo óptimo. Durante todo este proceso, se pueden emplear varios algoritmos de oversampling, entre los cuales destacan ADASYN, SMOTE, SVMSMOTE, entre otros.

Un aspecto destacado de Python es la disponibilidad de bibliotecas preprogramadas, como la mencionada biblioteca *imbalanced-learn* de Lemaître et al. (2017), que incluye implementaciones de estos algoritmos y otros, por consiguiente, los datos deben ajustarse a los parámetros específicos requeridos por esta biblioteca para su correcta aplicación.

5.4.2.3 Tratamiento preliminar para entrenar las redes neuronales.

Una vez obtenido la base de datos final, se debe reescalar los datos para garantizar un rendimiento óptimo y la estabilidad del modelo con ayuda de las **ecuaciones 2** y **5** y finalmente, antes de entrenar los modelos de redes neuronales, es necesario realizar un tratamiento preliminar mediante la división del conjunto de datos en conjuntos de entrenamiento y validación. Para llevar a cabo esta tarea, empleamos la biblioteca *scikit-learn*, desarrollada por Pedregosa et al.

(2011), y específicamente la función *train_test_split*; esta función no solo facilita la división de datos y selección del porcentaje deseado de la división, sino que también ofrece la opción de barajar los datos, esta práctica resulta beneficiosa para evitar sesgos y mejorar el rendimiento del modelo.

5.4.2.4 Aplicación de deep learning.

Para aplicar algoritmos de aprendizaje profundo, es fundamental tener un conocimiento sólido del álgebra y programación. Python, como lenguaje de programación, dispone de bibliotecas que pueden instalarse en diversos entornos, en el ámbito de la inteligencia artificial, destacan tres bibliotecas: *scikit-learn, tensorflow* y *PyTorch*.

Como se detalló en secciones anteriores, el uso *scikit-learn* está orientada al machine learning, pero también se utiliza para diversas funciones, como la división de conjuntos de datos y el escalado, entre otras, que se emplearon en este trabajo. Sin embargo, para el desarrollo de las redes neuronales, se optó por la biblioteca *PyTorch* de Paszke et al. (2019), el cual es una herramienta de código abierto con amplia documentación disponible en línea.

A continuación, se presenta el código para la implementación de las tres distintas redes neuronales, siendo necesario destacar que, para ingresar datos y construir las redes neuronales, se aprovechan las diversas funciones proporcionadas por *PyTorch*, como se ilustra en la **Figura 25**.

```
from torch.utils.data import TensorDataset, DataLoader
import torch.nn as nn
import torch.optim as optim
```

Figura 25. Paquetes para cargar datos en las redes neuronales y para crear la red neuronal.

5.4.2.4.1 RNN, LSTM y GRU.

La construcción de las redes sigue un enfoque específico: se define una clase para cada tipo de red, la cual cuenta con la arquitectura RNN, LSTM o GRU y la salida de cada una de estas redes se conectan a una capa lineal y una función de activación sigmoide. Además, la función *batch_first* especifica que la matriz de datos debe tener la forma *(batch_size, sequence_length, input_size)*. Esta elección se realiza para facilitar el manejo de los datos sin necesidad de ajustes adicionales en la dimensionalidad de la matriz de datos.

En las **Figuras 26, 27 y 28**, se presentan los códigos utilizado para crear los tres modelos de redes neuronales: RNN, LSTM y GRU, respectivamente.

```python
class RNNModel(nn.Module):
    def __init__(self, input_dim, hidden_dim, layer_dim, output_dim, dropout_prob):
        super(RNNModel, self).__init__()
        self.hidden_dim = hidden_dim
        self.layer_dim = layer_dim
        self.rnn = nn.RNN(input_dim, hidden_dim, layer_dim, batch_first=True,
                          dropout= dropout_prob)
        self.fc = nn.Linear(hidden_dim, output_dim)
        self.final = nn.Sigmoid()

    def forward(self, x):
        h0 = torch.zeros(self.layer_dim, x.size(0),self.hidden_dim).requires_grad_().to(device)
        out, h0 = self.rnn(x,h0.detach())
        out = out[:,-1,:]
        out = self.fc(out)
        out =  self.final(out)

        return out
```

Figura 26. Código Python de red neuronal RNN, con la biblioteca *PyTorch*.

En las diversas figuras se aprecia el empleo de los módulos *nn.RNN*, *nn.LSTM* y *nn.GRU* proporcionados por **PyTorch**. Estas tres arquitecturas de redes neuronales recurrentes están preprogramadas en la biblioteca, y para utilizarlas solo es necesario proporcionar los parámetros específicos de cada arquitectura, dado que estas redes pertenecen a la misma familia, comparten similitudes en sus parámetros, razón por la cual se ingresan las mismas variables en las tres instancias.

```python
class LSTMModel(nn.Module):
    def __init__(self, input_dim, hidden_dim, layer_dim, output_dim,
                 dropout_prob):
        super(LSTMModel, self).__init__()
        self.hidden_dim = hidden_dim
        self.layer_dim = layer_dim
        self.lstm = nn.LSTM(input_dim, hidden_dim, layer_dim, batch_first = True,
                            dropout = dropout_prob)
        self.fc = nn.Linear(hidden_dim, output_dim)
        self.final = nn.Sigmoid()

    def forward(self, x):
        h0 = torch.zeros(self.layer_dim, x.size(0), self.hidden_dim).requires_grad_().to(device)
        c0 = torch.zeros(self.layer_dim, x.size(0), self.hidden_dim).requires_grad_().to(device)
        out, (hn, cn) = self.lstm(x, (h0.detach(), c0.detach()))
        out = out[:,-1,:]
        out = self.fc(out)
        out =  self.final(out)

        return out
```

Figura 27. Código Python de red neuronal LSTM, con la biblioteca *PyTorch*.

También se puede notar el uso del parámetro *dropout*, el cual se emplea para evitar el sobreajuste y prevenir que la red aprenda ruido o peculiaridades de los datos, el no usar hiperparámetros puede resultar en un rendimiento deficiente en datos no vistos e incluso afectar al TL.

```python
class GRUModel(nn.Module):
    def __init__(self, input_dim, hidden_dim, layer_dim, output_dim, dropout_prob):
        super(GRUModel, self).__init__()
        self.layer_dim = layer_dim
        self.hidden_dim = hidden_dim
        self.gru = nn.GRU(
            input_dim, hidden_dim, layer_dim, batch_first=True, dropout=dropout_prob
        )
        self.fc = nn.Linear(hidden_dim, output_dim)
        self.final = nn.Sigmoid()



    def forward(self, x):
        h0 = torch.zeros(self.layer_dim, x.size(0), self.hidden_dim).requires_grad_().to(device)
        out, _ = self.gru(x, h0.detach())
        out = out[:, -1, :]
        out = self.fc(out)
        out = self.final(out)

        return out
```

Figura 28. Código Python de red neuronal GRU, con la biblioteca *PyTorch*

Finalmente, se agregan las funciones de optimización, pérdida e hiperparámetros faltantes.

5.4.2.5 Metrícas de evaluación de desempeño.

Cuando se entrena una red neuronal siempre se debe evaluar como esta red aprende, por ende se hace uso de diferentes métricas como: accuracy, precision, specifity, recall, F1-score; las cuales se detallan en el marco teórico con las **ecuaciones 19, 22, 21, 20** y **23**; y dado que el TL2 se trabaja con datos desbalanceados se hace uso solo del F1-score Pan y Yang (2010).

En la **Figura 29** observamos el código para evaluar el modelo, es importante destacar que este proceso solo es para evaluar el modelo, para el entrenamiento también se puede evaluar el desarrollo de las métricas, pero en nuestro caso solo hacemos uso de accuracy y la función de pérdida.

```python
from sklearn.metrics import precision_score, recall_score, roc_curve, auc, roc_auc_score, f1_score
import numpy as np
import matplotlib.pyplot as plt

def metrics(predictions, values):
    predictions = np.array(predictions).reshape(-1,1)
    values = np.array(values).reshape(-1,1)
    y_predict = np.where(predictions>0.5,1,0)

    print('Test Accuracy -',round(accuracy_score(values,y_predict),3))
    print('Test Precision -',round(precision_score(values,y_predict),3))
    print('Test Specifity',round(recall_score(values, y_predict, pos_label=0),3))
    print('Test Recall -',round(recall_score(values,y_predict),3))
    print('f1-score', round(f1_score(values, y_predict, average = 'binary'),3))

    fpr_v, tpr_v, _ = roc_curve(values,predictions)
    roc_auc_v = auc(fpr_v, tpr_v)

    plt.plot(fpr_v, tpr_v, 'b', label='Test AUC = %0.4f'%roc_auc_v)
    plt.legend(loc='lower right')
    plt.plot([0,1],[0,1],'r--')
    plt.xlim([0,1])
    plt.ylim([0,1])
    plt.ylabel('True Positive Rate')
    plt.xlabel('False Positive Rate')
    plt.show()
```

Figura 29. Código Python para el cálculo de las diferentes métricas.

5.4.2.6 Transfer learning (TL).

Como se observa en el flujograma (**Figura 19**), se realizan dos tipos de transfer learning (TL1 y TL2) con diferentes estrategias.

Simani et al. (2023) propone 5 estrategias, de las cuales se opta por usar 3 las cuales son:

- Estrategia 1: Se transfiere el aprendizaje del modelo inicial a otro dominio, pero bloquea las capas iniciales del modelo inicial y se reentrena únicamente las últimas capas del modelo, modificando solo los pesos de las capas no bloqueadas, y no se modifica la estrutura de la red (Ver **Figura 30**).

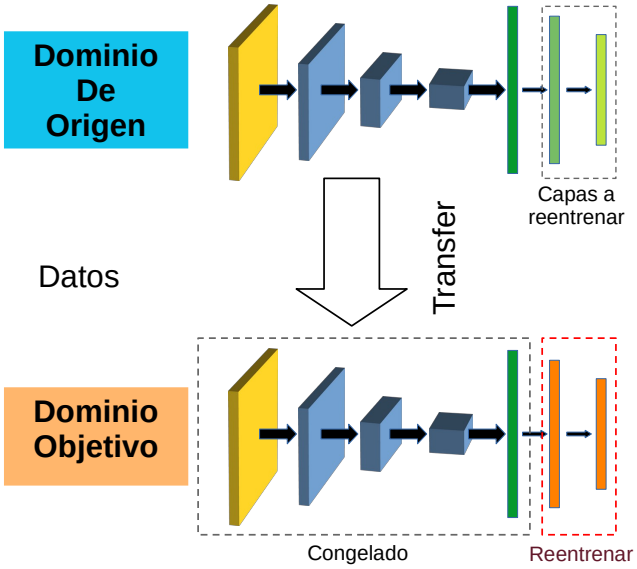


Figura 30. Estrategia 1 de TL.

Fuente: Simani et al. (2023).

- Estrategia 2: Se transfiere el aprendizaje del modelo inicial a otro dominio reentrenando todas las capas, modificando todos sus pesos, pero la estructura de la red se mantiene igual (Ver **Figura 31**).

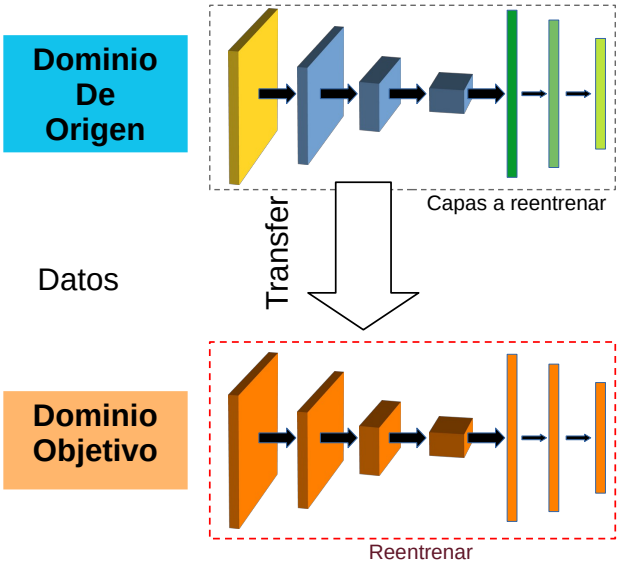


Figura 31. Estrategia 2 de TL.

Fuente: Simani et al. (2023).

- Estrategia 3: Se transfiere el aprendizaje del modelo inicial a otro dominio sin ninguna modificación en sus pesos, ni la estructura de la red, como se muestra en la **Figura 32**.

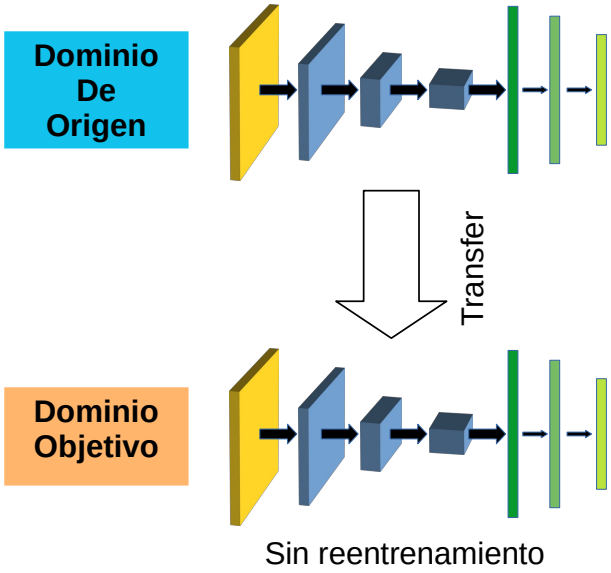


Figura 32. Estrategia 3 de TL.

Fuente: Simani et al. (2023).

5.4.2.6.1 Transfer Learning 1 (TL1).

Para el primer TL1 se considera la estrategia 3, el proceso se lo realiza de la siguiente manera:

- Tomamos el dataset B (ver **Figura 21**) y seleccionamos los intervalos.
- Cargamos el modelo base entrenado.
- Aplicamos el TL con la estrategia 3, o en términos más simples, evaluamos el modelo con el intervalo tomada del dataset B y verificamos los resultados.
- Gráficamos la curva de predicciones.

5.4.2.6.2 Transfer Learning 2 (TL2).

En el contexto del TL2, se tienen en cuenta las estrategias 1, 2 y 3 propuestas por Simani et al. (2023). Debido a que el TL2 se centra en predecir únicamente cuando el sistema SCADA registro un paro por fallo, se realiza un preprocesado de datos.

Dado que los datos han sido previamente procesados, el paso actual implica únicamente un proceso de reetiquetado, cuyo algoritmo se presenta en la **Figura 33**.

Algoritmo 3: Retiquetado de datos

Datos: Alarmas $\{A_{(I)}\}_{j=1}^{J}$,Modo de paro $\{M_{(Ts)}\}_{j=1}^{J}$, registro de la WT $\{T_{(t)}\}_{j=1}^{J}$ donde J es el número de WTs, t la hora y fecha que registo el sistema SCADA e I es un intervalo de tiempo, ts tiempo en que se registra el modo de paro y WTB es el WT utilizado para el modelo base.

Resultados: Columna de etiquetas para cada registro $\{L_{(t)}\}_{j=1}^{J}$.

Buscamos la forma de paro de interes $\{M_{(Ts)}\}_{j=1}^{J} = fault\ stop$ con $j \neq WTB$.

Identificamos la fecha y hora que inicio la alarma t_i y la hora y fecha que se finalizo la alarma t_f, con $I = t_f - t_i$.

Para cada Alarma $A_{(I)}^{j}$ en el tiempo ts, con $j \neq WTB$ **hacer**:

    Si $A_{t_i}^{j} \leq M_{ts}^{j} \leq A_{t_f}^{j}$ y $t_i \leq ts \leq t_f$:

        Se va concatenando $\{A_{(I)}\}^{j}$

    Caso contrario:

        Se eliminan los datos

    Se repite todo el ciclo para cada instante ts del StopModoQuery para cada Wt j

Del ciclo anterior tenemos un nuevo $\{A_{(I)}\}_{j=1}^{J}$ solo con paros de interes ($fault\ stop$).

Para cada registro de la WT T^{j} con $j \neq WTB$, en el tiempo t **hacer**:

    Si $A_{t_i}^{j} \leq T_{t}^{j} \leq A_{t_f}^{j}$ y $t_i \leq t \leq t_f$:

        $L_{t}^{j} = \mathbf{1}$

    Caso contrario:

        $L_{t}^{j} = \mathbf{0}$

    L_{t-1}^{j} se concatena el nuevo registro L_{t}^{j}

    Se repite todo el ciclo para cada instante t del sistema SCADA y para cada WT j

Finalmente se adjunta la matriz L al registro T como una columna llamada “label”.

Figura 33. Retiquetado de datos, solo cuando existe un paro por fallo de WT.

Una vez que se cuenta con la base de datos con el reetiquetado, se eligen tres WTs y una muestra de datos de un año de cada WT, a los cuales se les aplica TL con las tres diferentes estrategias. Debido a que los datos se encuentran desbalanceados se emplea el F1-score para evaluar el desempeño de cada estrategia.

Una vez obtenido los resultados de aplicar el TL se procede a reajustar los hiperparámetros con la finalidad de mejorar el valor del F1-score hasta lograr alcanzar un máximo posible, de cada estrategia y el WT que logre el mejor resultado se guarda el modelo para luego aplicar TL a los datos del WT con el que se creo el modelo base.

5.4.3 OE3. Desarrollar un dashboard para el manejo y visualización de los resultados obtenidos

La implementación del dashboard se lleva a cabo mediante el uso de bibliotecas de Python, en particular, se utiliza la biblioteca *Plotly*. Dentro de su entorno, se aprovecha el paquete denominado *Dash Open Source*, el cual está diseñado para la creación de dashboards sin requerir JavaScript. Estos dashboards pueden ejecutarse localmente en el servidor localhost, sin embargo, en caso de necesitar un servidor público, se tiene la opción de adquirir *Dash Enterprise* mediante el pago correspondiente, una vez que nuestro dashboard en Dash está listo, puede ser desplegado en un servidor local o público (Plotly Technologies Inc., 2015).

En el proceso de desarrollo, se importan todas las funciones y base de datos esenciales, es importante destacar que internamente se trata de una página web, por lo tanto, es necesario emplear cierta programación en *HTML* y *CSS*, la **Figura 34**, se muestra la importación de los paquetes fundamentales necesarios para la creación del dashboard.

```
from dash import html, dcc, callback, Output, Input, State, dash_table, callback_context
import dash_mantine_components as dmc
from dash_iconify import DashIconify
from app import app
import pandas as pd
import plotly.express as px
import plotly.graph_objects as go
import dash
import numpy as np
import matplotlib
matplotlib.use('agg')
import matplotlib.pyplot as plt
import base64
from io import BytesIO
```

Figura 34. Importación de paquetes dash, plotly y html, para el desarrollo del dasboard.

Mediante las funciones disponibles, se utilizan widgets, tablas y se posicionan según nuestras necesidades, también se crean figuras tanto estáticas como interactivas, además, para optimizar el procesamiento y la recuperación de datos que se presentarán en el dashboard, se emplea otra biblioteca que interactúa con una base de datos, en nuestro caso, hemos optado por utilizar *pandas* para cargar archivos *.csv*.

6. Resultados

En esta sección, se presentan los resultados obtenidos la aplicación del Transfer Tearning para la detección de fallas en aerogeneradores utilizando datos operativos y de alarmas del sistema SCADA, basado en la metodología previamente expuesta, y se organizan de manera sistemática y clara en una interfaz gráfica, desarrollada para la visualización de los datos y resultados. Esta sección proporciona una visión detallada y comprensible de los resultados obtenidos, facilitando su interpretación y análisis.

6.1 Implementar Data Mining al registro histórico del periodo 2014-2021 de los datos SCADA y el registro de alarmas de fallos del parque eólico Villonaco del cantón Loja

Tras la conversión de los datos a archivos .csv mediante Python, se procedió a realizar un análisis utilizando técnicas de estadística descriptiva, con el objetivo de obtener una visión general del estado de los datos. Dada la dimensionalidad del conjunto de datos, se presenta en la **Tabla 6** una muestra representativa que incluye únicamente cuatro columnas. Este enfoque se adopta para resaltar aspectos claves y facilitar la interpretación de las observaciones, considerando la complejidad del dataset.

Tabla 6. Indicadores estadística de algunas variables de estudio.

	wind_speed_max	grid_active_power_min	converter_reactive_power_avg	converter_reactive_power_min
count	386975.00	386975.00	386975.00	386975.00
mean	13.880797	671.998067	0.092258	-20.740309
min	0	-539.50	-11.22	-1750.05
25 %	9.44	128.04	-0.030	-22.21
50 %	14.27	629.99	0	-11.1
75 %	18.10	1151.57	0.04	-5.12
max	40.05	1557.91	14.45	5.12
std	6.278244	543.812486	1.709758	34.6514

Posterior al análisis estadístico inicial de nuestros datos, se procedió a examinar la distribución de las variables a lo largo de los siete años. En la **Figura 35** se identificó la presencia de valores faltantes en todos los WTs. Específicamente, se observó que el WT11 exhibe una mayor incidencia de datos faltantes, seguido por el WT10 y WT9, que también presentan ausencia de datos en algunos sectores. Por otro lado, el resto de los WTs solo muestran ausencia de datos en el periodo comprendido desde enero de 2021 hasta aproximadamente mayo de 2021.

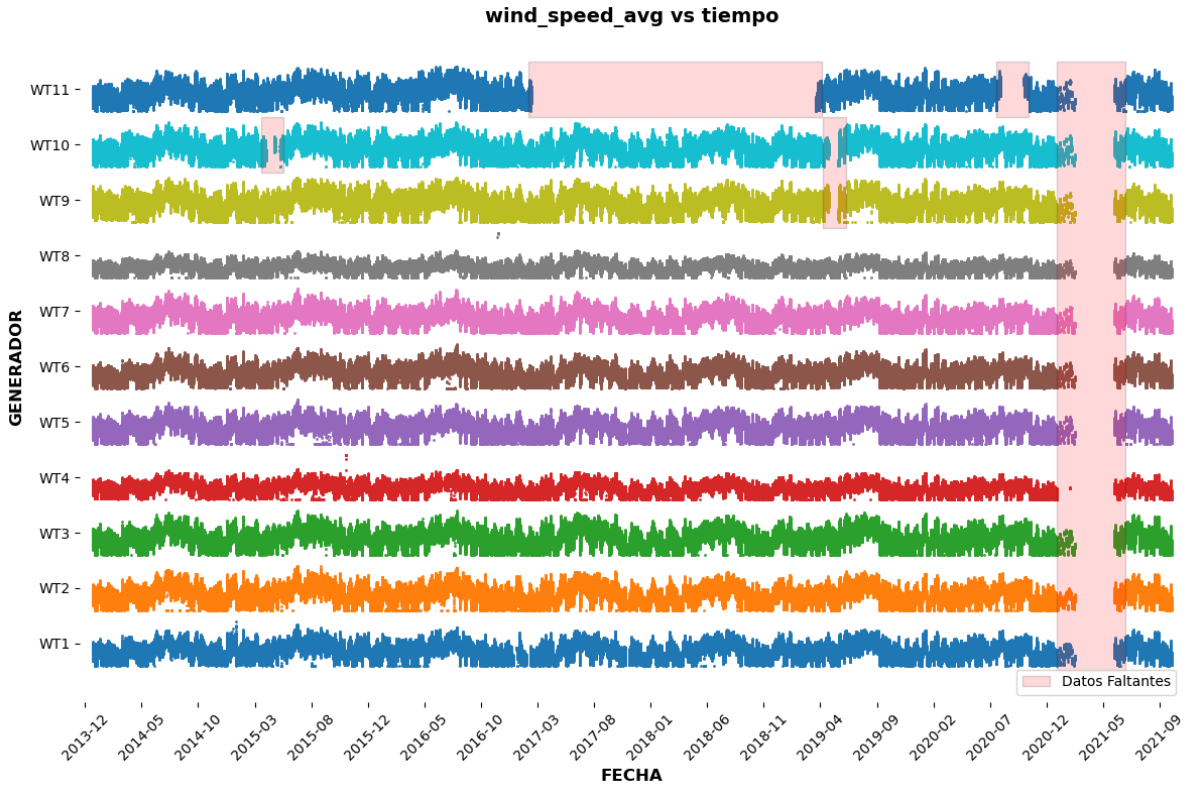


Figura 35. Distribución de datos durante los 7 años variable wind_speed_avg.

Habiendo identificado el mencionado problema, se constató que la presencia de datos faltantes constituía un porcentaje mínimo en relación con el total de los datos; por consiguiente, se decidió prescindir de la aplicación de cualquier método de imputación de datos, optando por trabajar con la totalidad de los datos disponibles para llevar a cabo el análisis de los datos denominados “NaN”.

En la **Figura 36**, se detallan las variables que presentan valores “NaN”, siendo necesario señalar que estos datos “NaN” representaron aproximadamente el 40 % de los datos de dichas variables, motivo por el cual se optó por eliminar las cuatro variables correspondientes del dataset. Como resultado, se obtuvo un nuevo conjunto de datos con dimensiones $(n, 68)$, donde n representa el total de registros acumulados durante los ocho años. A partir de esta etapa, se llevó a cabo el etiquetado de los datos.

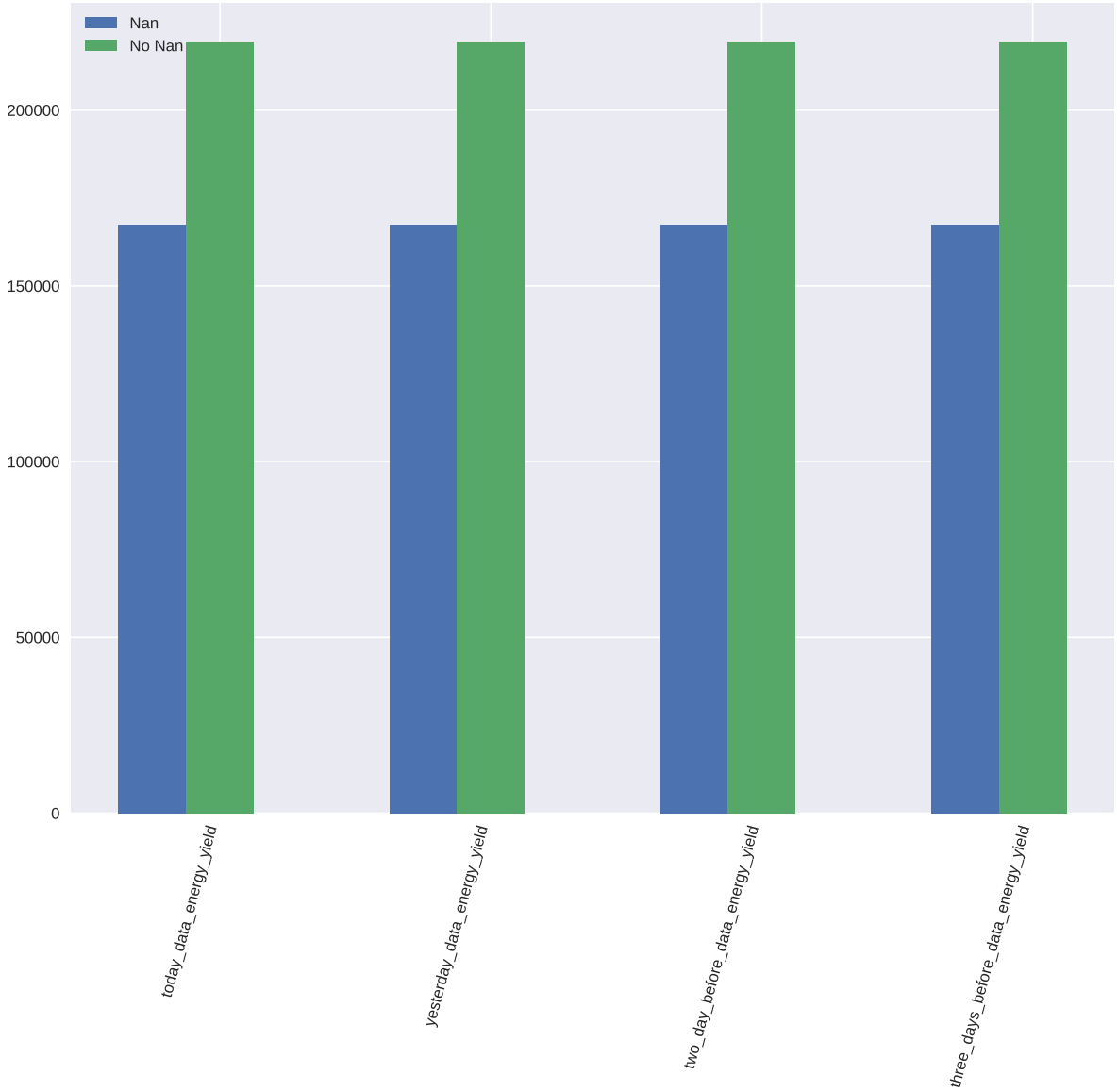


Figura 36. Muestra datos “NaN” para el WT5.

La primera fase de etiquetado de datos reveló una marcada desproporción en la distribución de las etiquetas. Como se aprecia en la **Figura 37**, el WT5 exhibe el mayor número de registros relacionados con alarmas (fallos, paros de emergencia, y otros eventos) con un total de 364055 etiquetas iguales a 0 y 22920 etiquetas iguales a 1. Por lo tanto, se decidió utilizar el WT5 para entrenar el modelo base, de esta manera dar cumplimiento el segundo objetivo específico de la investigación.

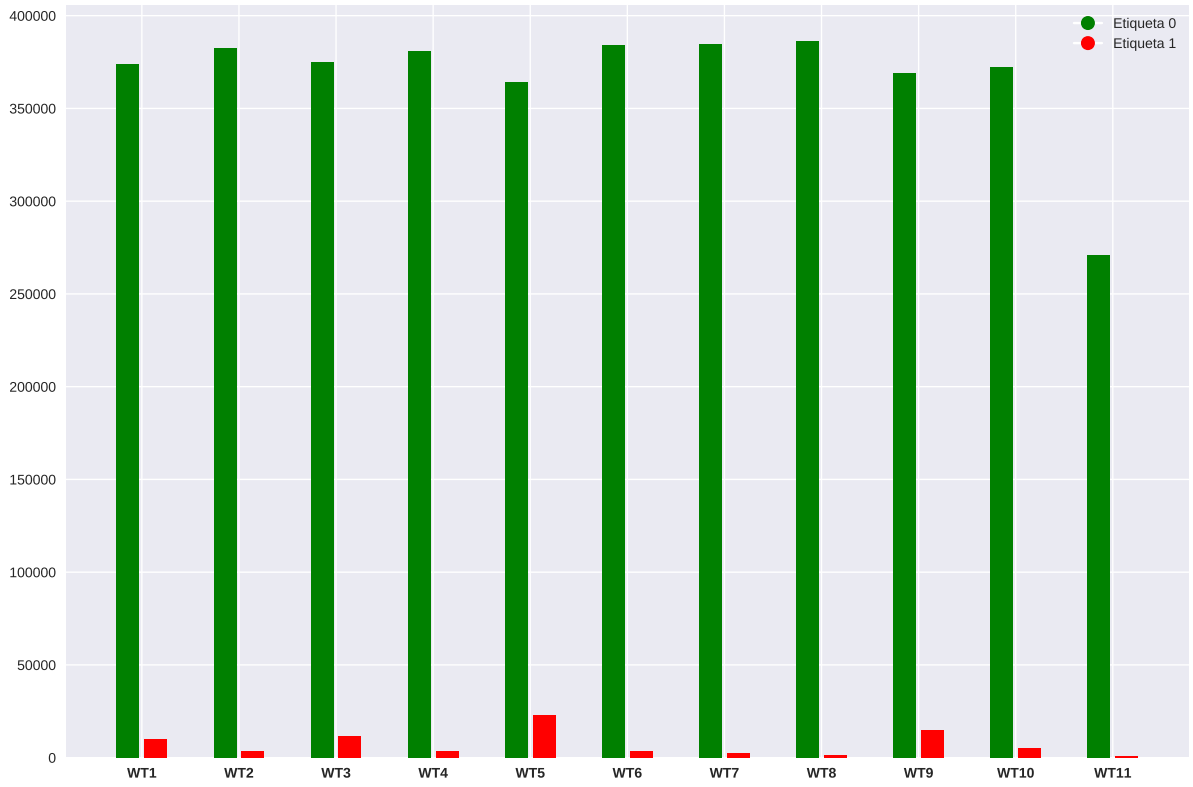


Figura 37. Cantidad de datos etiquetados de las WTs.

Con los datos del WT5, se llevó a cabo un análisis de las variables, debido a que el trabajar con las 68 variables podría resultar en un costo computacional considerablemente alto; adicionalmente, considerando que existe la posibilidad de colinealidad y redundancia entre algunas variables, por ejemplo, aquellas que representan el valor máximo, mínimo y promedio.

Para la selección de variables se aplica el método VIF, teniendo como resultado 20 variables que tienen un VIF menor a 5, lo que implica que su colinealidad es débil o moderado, en la **Tabla 7** observamos las variables seleccionadas. Es importante mencionar que para tener un mejor control de los datos y gráficos también se considera la variable “timestamp” que proporciona la fecha y hora del registro pero no se usa para entrenar el modelo, adicionalmente se selecciono la variable “label” dado que indica si el WT está en fallo o funcionamiento normal la cual sirve para entrenar el modelo. Las mismas variables se va a usar en todo el proceso de DL y TL.

Tabla 7. Variables seleccionadas para entrenar el modelo base y evaluar los diferentes estrategias de TL.

	Variable
1	wind_speed_max
2	generator_capacitors_temperature_max
3	converter_reactive_power_avg
4	converter_reactive_power_max
5	converter_reactive_power_min
6	ambient_temperature_min
7	blade_angle_max
8	nacelle_position_avg
9	grid_U2_avg
10	acceleration_nacelle_max
11	nacelle_temperature_max
12	rectifier_temperature_max
13	chopper_igbt_temperature_max
14	dc_inductor_temperature_max
15	pitch_motor_temperature_3_max
16	pitch_capacitor_temperature_2_max
17	data_energy_yield
18	data_consumed_energy_yield
19	data_service_time
20	data_grid_control_standstill_time

Con la selección de variables concluida, finaliza la fase de data mining. No obstante, en los procesos subsiguientes de TL, se implementan etapas adicionales de preprocesado y reetiquetado.

6.2 Aplicar transfer learning para predicción de fallas y evaluar su rendimiento mediante diferentes métricas

6.2.1 Transfer learning 1 (TL1)

En el contexto del primer transfer learning, se han delimitado las regiones A, B (normales) y C (fallo) a partir de la base de datos del WT5. Esta segmentación resulta en un total de 15 intervalos identificados en la región B, mientras que el resto de los datos se asigna a las regiones A y C, según sus respectivas etiquetas. En la **Figura 38** destaca visualmente los datos de la región B en color amarillo, mientras que las regiones A-C se resaltan en verde. Además,

la representación de la distribución de las etiquetas a lo largo de los 8 años se muestra en color magenta.

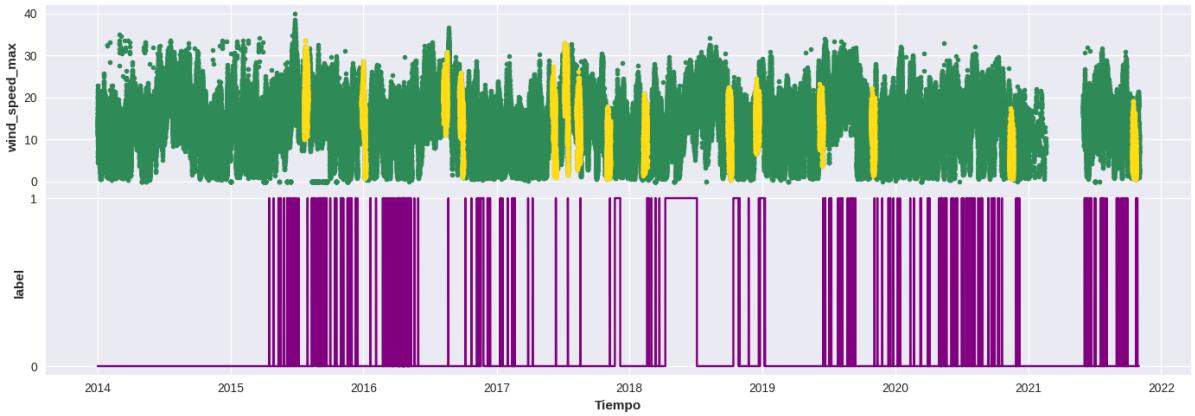


Figura 38. Curva distribuición wind_speed_max, label vs tiempo.

A partir de los datos de las regiones A, B y C; se realizó un recuento de todas las etiquetas por cada región, revelando un evidente desbalance (**Figura 39**). En la región A-C, se observa un total de 341112 etiquetas 0 y 22943 etiquetas 1, mientras que en la región B se muestran 21506 etiquetas 0. Como respuesta a este desbalance, se implementó un algoritmo de oversampling en las regiones A-C, en la región B no se aplica ningún algoritmo dado que se usarán para el TL1.

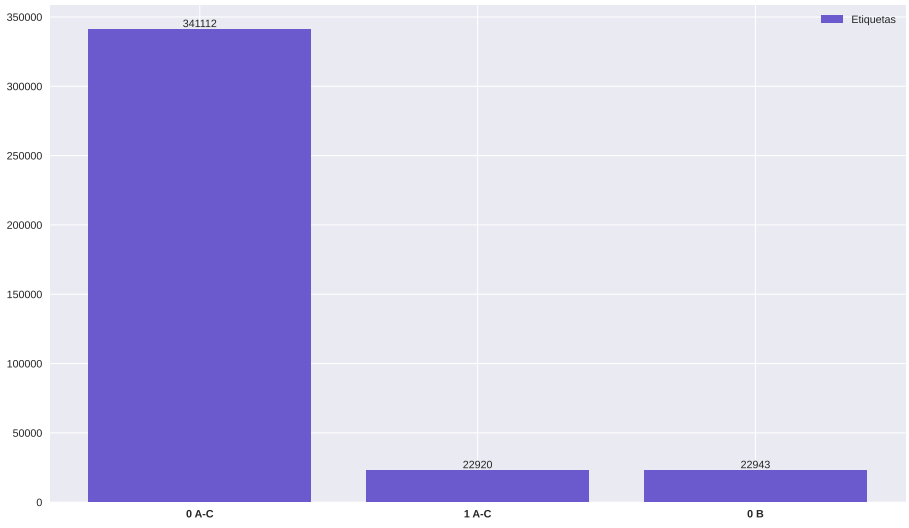
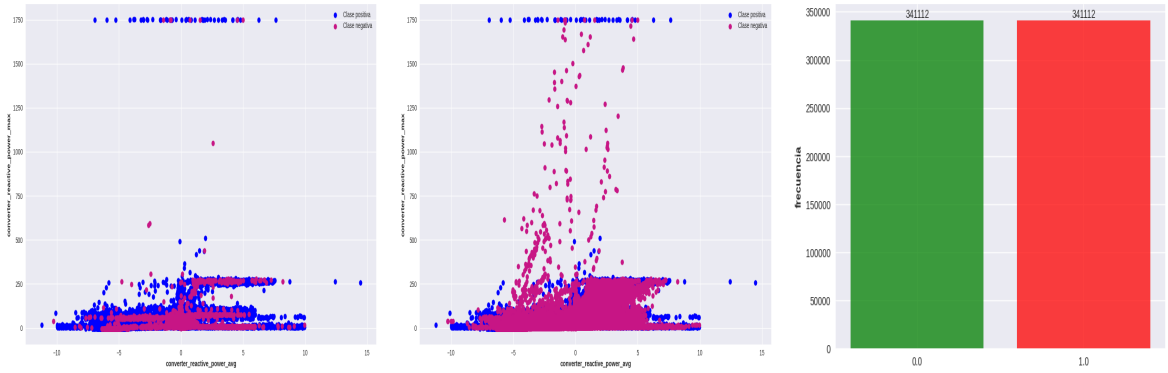


Figura 39. Frecuencia de etiquetas en las regiones A-B-C.

Para este estudio, se eligió el algoritmo *SMOTE*, logrando así una relación equilibrada de 1:1 en la región correspondiente. La **Figura 40** ilustra los datos antes y después de la aplicación de este algoritmo, donde **a)** es la distribución de converte_reactive_power_avg vs

converte_reactive_power_max sin la implementación del algoritmo, **b)** muestra la distribución después de aplicar oversampling, destacando cómo los puntos magenta se han incrementado en toda la figura, y finalmente **c)** resenta la distribución final donde ambas etiquetas cuentan con la misma cantidad de datos.



(a) (b) (c)

Figura 40. Resultados de algoritmo de oversampling. a) No oversampling, b) Oversampling, c) Total etiquetas.

Con los datos balanceados, se procedió a dividir el conjunto de datos en entrenamiento y evaluación, utilizando el 80 % para el primero y el 20 % para el segundo. Posteriormente, se realizaron ajustes en los hiperparámetros y la estructura de las redes RNN, LSTM y GRU. Es importante destacar que los ajustes de los hiperparámetros son uniformes para todas las redes, permitiendo así una evaluación comparativa de sus rendimientos.

En la **Tabla 8**, se presentan los valores específicos asignados a dichos hiperparámetros.

Tabla 8. Configuración de hiperparámetros de las redes RNN, LSTM y GRU.

Hiperparámetro	valor
input_dim	20
output_dim	1
hidden_dim	89
layer_dim	2
dropout	0.01
n_epochs	100
learning_rate	1×10^{-3}
weight_decay	9.9×10^{-6}
fun. perdida	MSELoss
optimizador	Adam

Durante el proceso de entrenamiento de los modelos, se procedió a examinar la evolución de la función de pérdida (Loss) durante las fases de entrenamiento y validación. Como se ilustra en la **Figura 41**, todos los modelos comienzan a converger a los 100 epochs con los datos de validación convergen en 0.3, mientras que la pérdida de los datos de entrenamiento converge en 0.02, por esta razón el epochs se fijó en dicho valor.

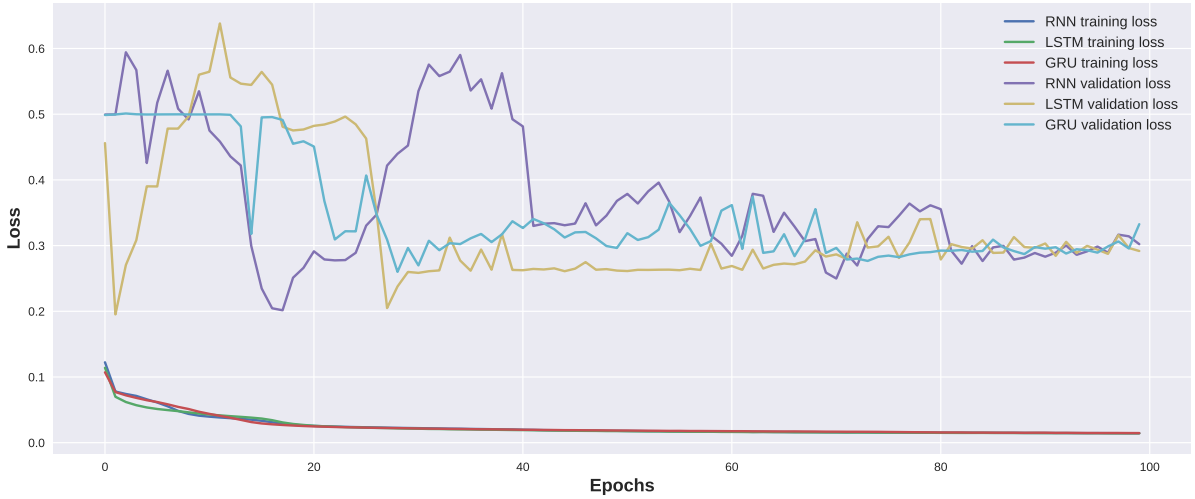


Figura 41. Curvas de las funciones de pérdida.

Posteriormente, se realizaron evaluaciones exhaustivas de los modelos utilizando los datos de validación y se registraron los resultados de diversas métricas de rendimiento, los cuales se detallan a continuación.

Tabla 9. Métricas de evaluación los modelos utilizados en el TL1.

Métrica	**RNN**	**LSTM**	**GRU**
Accuracy	0.69	0.704	0.663
Precision	0.643	0.658	0.617
Specifity	0.528	0.559	0.471
Recall	0.852	0.849	0.855
F1-score	0.733	0.741	0.717
Tiempo	12.524 [min]	10.826 [min]	12.01 [min]
Promedio	0.68955	0.7024	0.6713

Adicionalmente, la **Figura 42** presenta de manera gráfica las curvas ROC y el área bajo la curva (AUC), proporcionando una evaluación visual del desempeño del modelo en el conjunto de prueba (test).

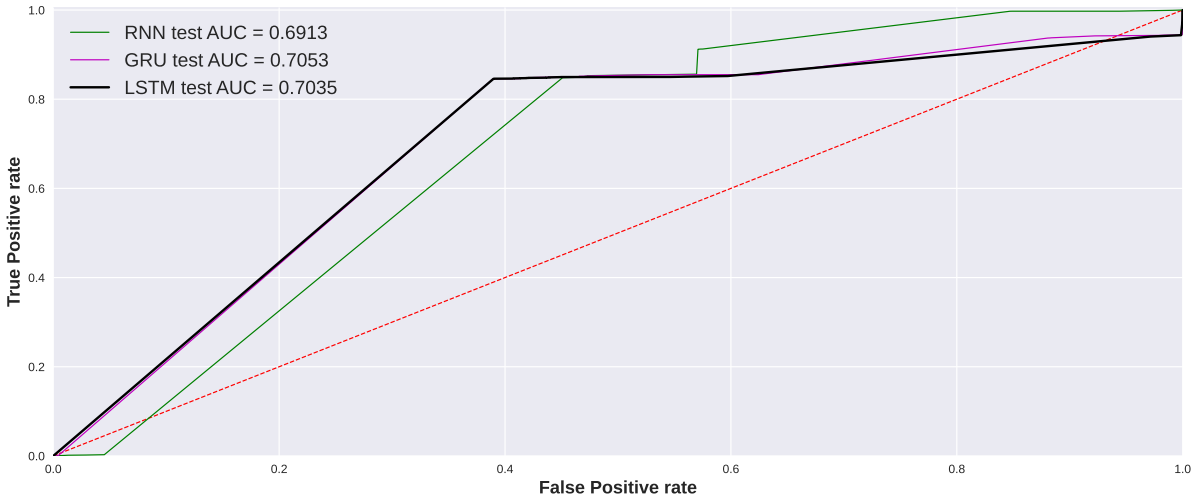


Figura 42. Conjunto de curvas ROC.

Con base en la información mostrada en la **Tabla 9**, se evidencia que, en promedio, la red LSTM logra un valor de 0.7024, superando a las redes RNN y GRU, que registran promedios de 0.6895 y 0.6713, respectivamente; además, la red LSTM presenta un tiempo de entrenamiento de 10.826 minutos, inferior a las otras redes que exceden los 12 minutos. Por ende, podemos afirmar que, para los propósitos de nuestra investigación, la LSTM destaca como la opción más eficiente y se designará como el modelo base para llevar a cabo los dos TL subsiguientes.

Tras la selección de la red LSTM, se inició la implementación del TL1, para este proceso, se retomaron los 15 intervalos correspondientes a la región B y se evaluó cada intervalo de dicha región utilizando el modelo base.

Los resultados de este análisis se presentan de manera gráfica en la **Figura 43**, destacando la capacidad del modelo para anticipar fallos o alarmas en los datos. Este paso representa una fase crucial en la aplicación del TL, donde se aprovechan los conocimientos previamente adquiridos por la LSTM para mejorar la anticipación de eventos en la región B

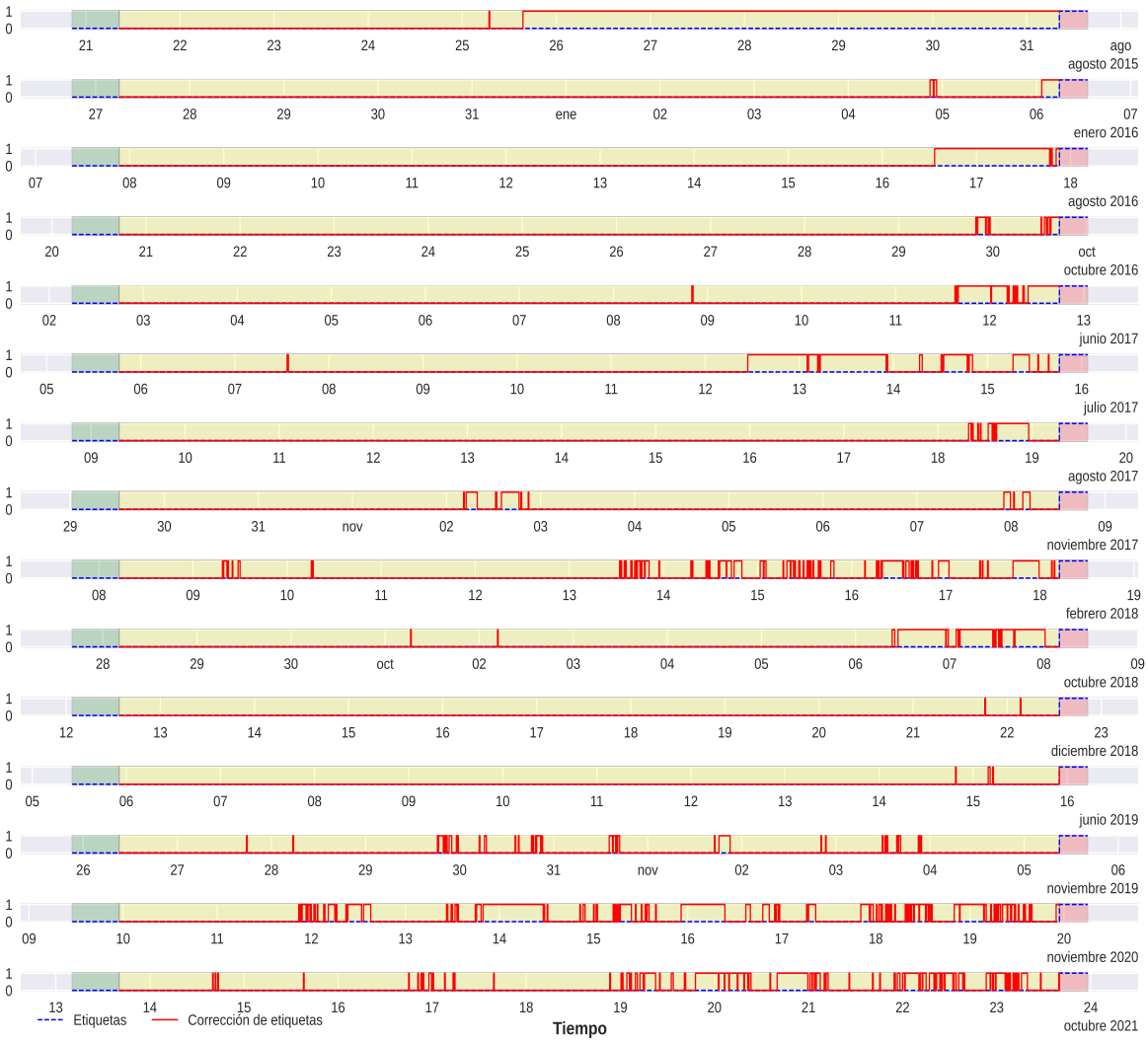


Figura 43. Resultados de aplicar el TL1.

En la **Figura 43** se puede observar las regiones A (verde), B (amarillo) y C (roja), el mes y año (parte derecha), los días(parte inferior de cada curva), y el comportamiento de las etiquetas; la línea roja es la etiqueta que predice el modelo, mientras la línea azul entre-cortada son las etiquetas reales.

Por ejemplo, en el 25 de agosto del 2015 se puede ver que se logra anticipar el fallo 5 días antes, donde primero se muestra el una alarma aproximadamente a las 10:00 horas; luego vuelvo a su funcionamiento normal para posteriormente volver a un estado de fallo el mismo día a las 17:00 horas y mantenerse en fallo hasta el instante que el sistema SCADA lo registra; así mismo, el 21 de diciembre del 2018, se registra una alarma aproximadamente a las 20:00 horas y luego retoma su estado normal hasta el 22 de diciembre que registra otra alarma a las 5:00 hora para finalmente retomar su estado normal hasta el momento que el sistema SCADA registra el fallo, siendo este el periodo más corto que anticipa un fallo de apenas dos diás antes.

En los diferentes intervalos se observa comportamientos similares y otros como en el caso de octubre del 2021 donde el modelo 9 días antes comienza a mostrar grupos de alarmas hasta que se registra el fallo así mismo por el sistema SCADA, este proceso se realiza con el fin de:

- Demostrar como se aplica la estrategia 3 a datos similares de los datos de entrenamiento.
- Que posteriormente se pueda realizar un ajuste de etiquetas dado que, como es conocido un fallo o alarma nunca se produce de un instante a otra como lo registra el sistema SCADA, siempre existe un tiempo previo donde inicia el problema y se acumula hasta generar un fallo o daño completo, de esta manera se podría reentrenar el modelo para mejorar las predicciones, pero este proceso ya no se considera dentro del trabajo de investigación.

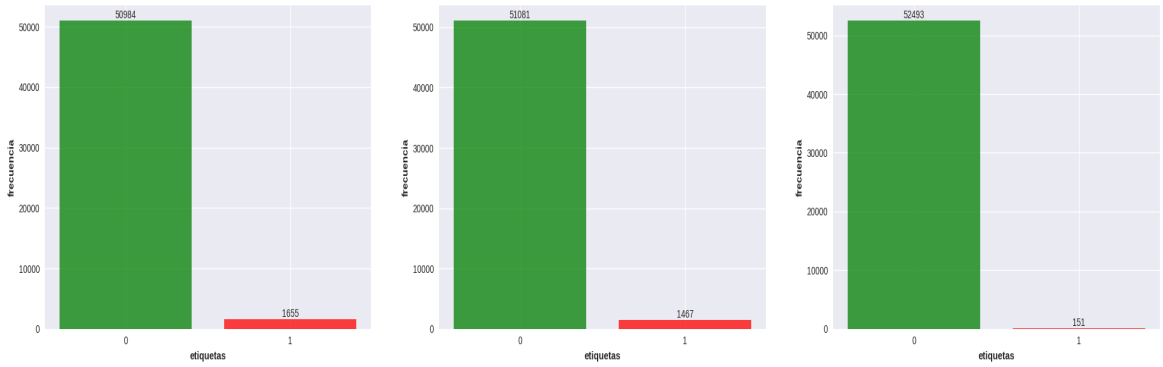
6.2.2 Transfer learning 2 (TL2)

Con el propósito de implementar el TL2, se optó por seleccionar otros WTs distintos al WT5. Esta elección se fundamenta en la consideración de la ubicación geográfica de las WTs y el hecho de que el modelo se entrenó inicialmente con datos del WT5.

Para asegurar que los nuevos datos abarquen variaciones y no se vean influenciados por la proximidad geográfica al WT5, se han seleccionado el WT1 y el WT11, los cuales se encuentran en los extremos de la CEV; adicionalmente, se incluyó la observación del comportamiento del modelo en un WT cercano, específicamente el WT6. Esta elección se realizó para obtener una perspectiva más completa sobre cómo el modelo se comporta en diferentes espacios geográficos y condiciones, este enfoque garantiza una diversidad geográfica en los datos de TL2, lo cual es esencial para evaluar la capacidad de generalización del modelo a través de diferentes ubicaciones de WTs.

En el proceso de etiquetado para el TL2, a diferencia del primer etiquetado donde se consideraron todos los registros de alarmas y fallos, en esta instancia solo se toman en cuenta los casos en los que se produce un paro por fallo (Fault stop). El procedimiento sigue una lógica similar y se detalla en la metodología explicada en el ácapite 5.4.2.6.2; además, en este caso específico, se limita la consideración a los registros del año 2016.

La **Figura 44** proporciona una visualización de la cantidad de etiquetas asignadas a cada WT, ofreciendo una perspectiva clara de la distribución de los eventos normales (0) y de paro por fallo (1) en cada WT. Este enfoque selectivo busca proporcionar información relevante y específica para la aplicación del TL2.



(a) WT1. (b) WT6. (c) WT11.

Figura 44. Total de etiquetas para cada WT.

La **Figura 44** revela un desbalance significativo en los datos, particularmente en el WT11, donde existen, 52493 etiquetas 0 y 151 etiquetas 1, dando lugar a una relación aproximada de 348:1. De manera similar, el WT1 presenta una relación de desbalance de 31:1, mientras que el WT6 muestra una relación de 35:1. A pesar de este desbalance, no se han aplicado procesos de oversampling o undersampling a los datos. Por lo tanto, se considera altamente recomendable emplear métricas diseñadas para abordar situaciones de desbalance en los datos, siendo la elección principal F1-score.

En el TL2 se implementaron tres estrategias distintas, entre las cuales se incluye la empleada en el TL1, detalladas a continuación:

- La estrategia 1: Dado que el modelo base cuenta con dos LSTM acopladas y una lineal de salida, se procedió a bloquear la LSTM1 y a reentrenar únicamente la LSTM2 y la capa lineal de salida.
- Para la estrategia 2: En esta estrategia, se realizaron entrenamientos en todas las capas del modelo utilizando los pesos previamente obtenidos del modelo base.
- La estrategia 3: Es crucial destacar que en este tipo de transferencia, solo se lleva a cabo la evaluación de nuevos datos, por lo que no se efectúan ajustes en los hiperparámetros ni en la estructura del modelo base.

Los hiperparámetros utilizados en el TL2 son idénticos a los empleados en el entrenamiento del modelo base (ver **Tabla 8**). Los resultados del TL2 evaluados únicamente con la métrica F1-score y los tiempos de entrenamiento para cada estrategia se presentan en la **Tabla 10**. Como podemos observar que, en la estrategia 1, los valores de F1-score para el WT1

y WT6 son bajos, sobre todo el WT1 con un F1-score de 0.11, mientras que para el WT11 se mantiene un rendimiento aceptable, incluso superando al modelo base.

Tabla 10. Resultados del TL2 sin reajuste de hiperparámetros.

WTs	Estrategia 1		Estrategia 2		Estrategia 3	
	F1-score	**t [seg]**	**F1-score**	**t [seg]**	**F1-score**	**t [seg]**
WT1	0.456	39.6	0.885	57	0.017	—
WT6	0.11	42.6	0.752	48	0.027	—
WT11	0.762	38.4	0.744	46.8	0.007	—

No aplica (—); Tiempo (t); Segundos (seg).

Esta diferencia se debe a que en la estrategia 1, solo se reentrenan ciertas capas de la red, lo que puede limitar su capacidad de adaptación a nuevos datos. En contraste, la estrategia 2 muestra resultados positivos para todos los WTs, al entrenar todas las capas a partir de los pesos del modelo base, se proporciona una mayor flexibilidad y capacidad de ajuste a los nuevos datos, resultando en métricas F1-score más favorables.

Es importante destacar que la estrategia 3, al ser una forma de transferencia de aprendizaje que solo evalúa los nuevos datos sin volver ajustar hiperparámetros ni estructura (no se puede reentrenar, por ende no se puede medir tiempos de entrenamiento), tiene limitaciones en la adaptación eficiente a los nuevos WTs por ende el F1-score no supera ni el 0.03 en los Wts.

Aunque la estrategia 2 podría considerarse como un resultado para demostrar la eficiencia del TL, se hizo un esfuerzo adicional para mejorar los resultados mediante el reajuste de hiperparámetros para las estrategias 1 y 2. Como resultado de estos reajustes, se logró un aumento en el F1-score, como se muestra en la **Tabla 11**. Para la estrategia 1, el WT11 aumentó de 0.762 a 0.78, lo que representa un incremento de aproximadamente el 2 %. Además, el WT6 experimentó un aumento de alrededor del 8 %, y el WT1 mostró un incremento significativo de aproximadamente el 22 %. En el caso de la estrategia 2, se alcanzaron valores notables con un F1-score de 0.9 para el WT1, 0.863 para el WT6 y 0.78 para el WT11, estos resultados reflejan un rendimiento excepcional del modelo en la predicción de paros por fallo, respaldado por una métrica F1-score muy alta.

Tabla 11. Resultados del TL2 con reajuste de hiperparámetros.

Parámetros	Estrategia 1			Estrategia 2		
	WT1	**WT6**	**WT11**	**WT1**	**WT6**	**WT11**
dropout	0,01	0,01	0,01	0,01	0,08	0,01
n_epochs	50	100	140	100	200	200
learning_rate	1×10^{-1}	1×10^{-2}	9×10^{-3}	1×10^{-3}	1×10^{-3}	1×10^{-3}
weight_decay	$9,4 \times 10^{-6}$	$9,9 \times 10^{-6}$	9×10^{-7}	$9,9 \times 10^{-6}$	$9,9 \times 10^{-6}$	$9,9 \times 10^{-6}$
F1-score	0,675	0,196	0,78	0,9	0,863	0,78
t [seg]	19,44	35,28	73,8	45,6	114	96,6

Tras obtener los resultados del TL2, se opta por la estrategia 2 y se emplean los pesos del WT1 para llevar a cabo una transferencia de aprendizaje al WT5 con el objetivo de mejorar los resultados obtenidos durante el diseño del modelo base. En este proceso, se carga el conjunto de datos correspondiente al WT5 y se procede a reentrenar el modelo sin la aplicación de algoritmos de oversampling, logrando un F1-score de 0.896 y un tiempo de entrenamiento de 10.8 minutos. Esta estrategia demuestra ser eficaz para la optimización del modelo base; y el aumento en el tiempo de entrenamiento se atribuye al incremento de las n_epochs en 200, manteniendo el resto de los hiperparámetros inalterados en comparación con el WT1 de la estrategia 2, como se detalla en la **Tabla 11**.

Es importante destacar la relevancia de los tiempos de entrenamiento obtenidos durante el TL2. Mientras que el modelo base requirió un tiempo considerable de 10,826 minutos, el TL2, tanto sin reajuste de hiperparámetros como con reajuste, se completaron en cuestión de segundos, considerando que se trata únicamente de una muestra correspondiente a un año. Cabe resaltar que el caso que demandó más tiempo fue la estrategia 2 en el WT6, incluso con reajuste de hiperparámetros, y aún así, se completó en 114 segundos y que a los datos no se aplicaron algoritmos de oversampling. Estos resultados ponen de manifiesto una significativa reducción en el tiempo de entrenamiento al aplicar el TL, subrayando la eficiencia de este enfoque para adaptar el modelo a nuevos conjuntos de datos. Además, destaca la capacidad de establecer un ciclo para mejorar los resultados, incluso para el modelo base.

6.3 Desarrollar un dashboard para el manejo y visualización de los resultados obtenidos

A continuación se describe las principales secciones del dashboard, se distinguen cuatro áreas principales: el home, procesado, Transfer Learning 1 y Transfer Learning 2 (**Figura 48**). Además, se incluyen imágenes de los logotipos de la UNL y CITE con hipervínculos que redirigen a sus respectivas páginas oficiales.

La sección del home (ver **Figura 45**), presenta el título de la investigación, la justificación, los objetivos y tres enlaces que permiten acceder al trabajo de titulación, al artículo científico y a un breve manual de cómo utilizar el dashboard; ádemas, se incluye una breve bibliografía de los autores con enlaces que redirigen a sus perfiles.

En la sección de procesado (ver **Figura 46**), se muestran los resultados del OE1, acompañados de funciones de graficado que facilitan la visualización de diversos datos iniciales; además, se presentan indicadores estadísticos, la cantidad de etiquetas por cada WT (justificando la selección de WT5) y una tabla que exhibe las variables seleccionadas para los procesos previamente explicados.

Figura 45. Ventana Home del dashboard.

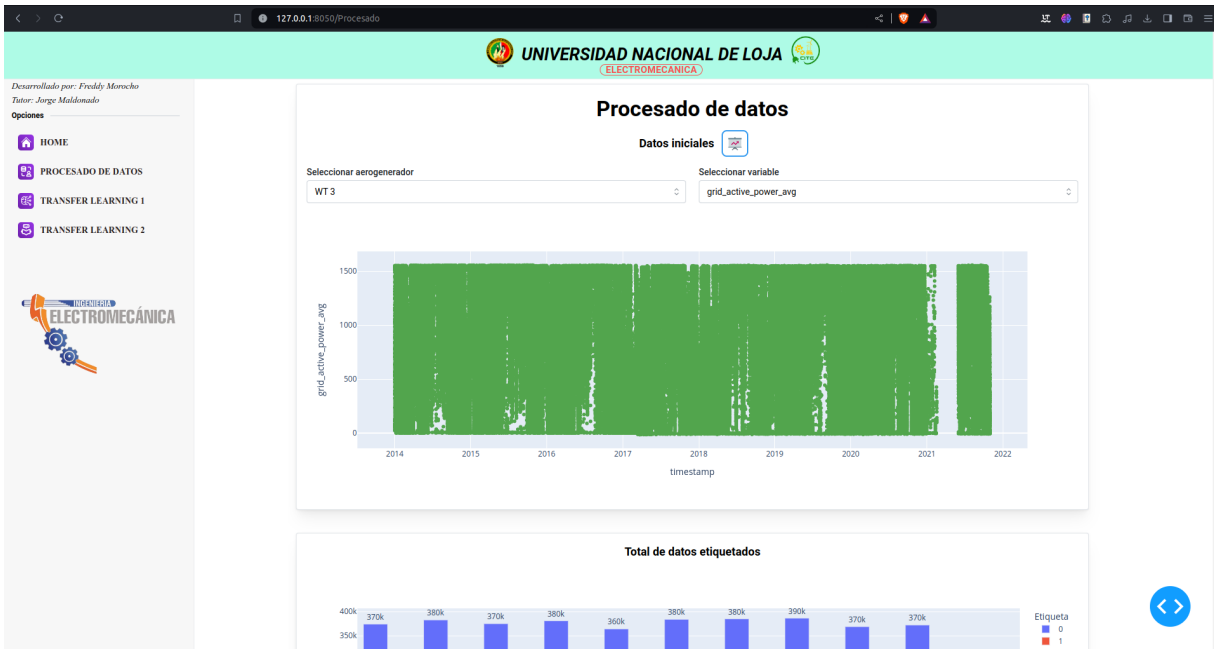


Figura 46. Ventana data mining del dashboard.

La sección de Transfer Learning 1 (ver **Figura 47**) incluye imágenes que explican la división de las regiones, cómo se distribuyen las etiquetas en cada región a lo largo de los 8 años del WT5 y cómo se ajustan los datos al aplicar el SMOTE. Además, presenta tablas que muestran los hiperparámetros de entrenamiento, la estructura de las redes y la evolución de la función de pérdida durante el entrenamiento. También se incorporan tablas acompañadas de gráficos de barras que visualizan de manera eficiente los resultados de las diferentes métricas, con un selector para explorar los resultados de TL1 en cada intervalo.

Por ultimo, la sección de Transfer Learning 2 (ver **Figura 48**), se muestran los WTs seleccionados y cómo cambian las etiquetas con el reetiquetado. Se explican las estrategias utilizadas y cómo cada una afecta los resultados. Se incluyen dos tablas que muestran los resultados del TL2 sin reajuste de hiperparámetros y con reajuste, junto con un selector para examinar el comportamiento del modelo en los datos de prueba.

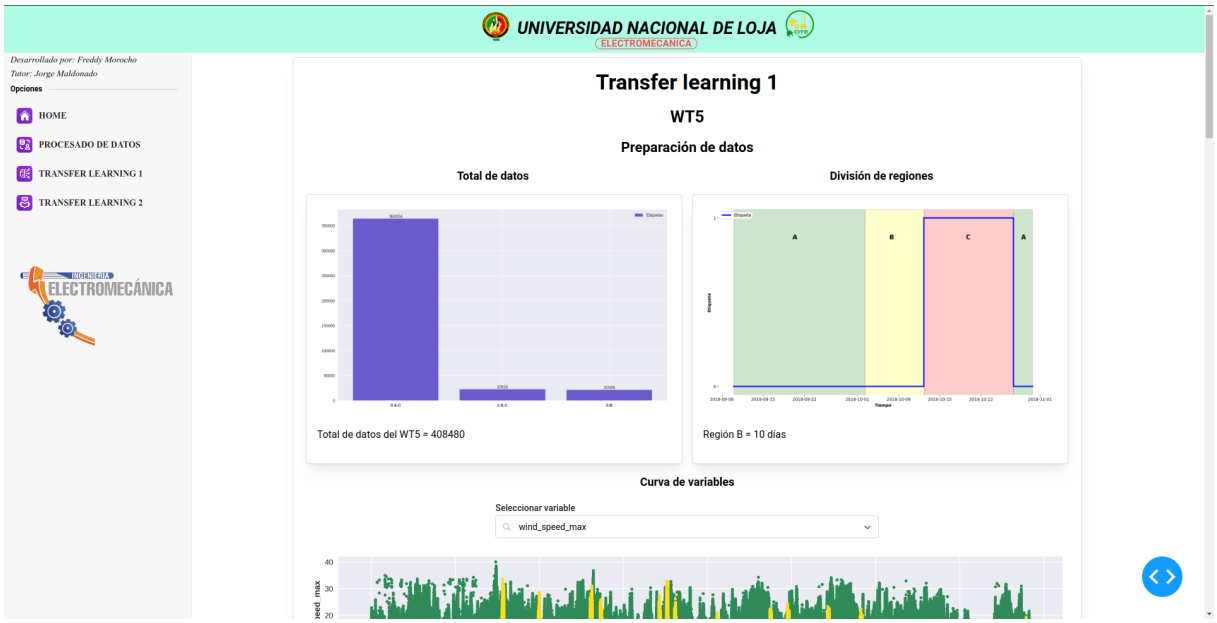


Figura 47. Ventana TL1 del dashboard.

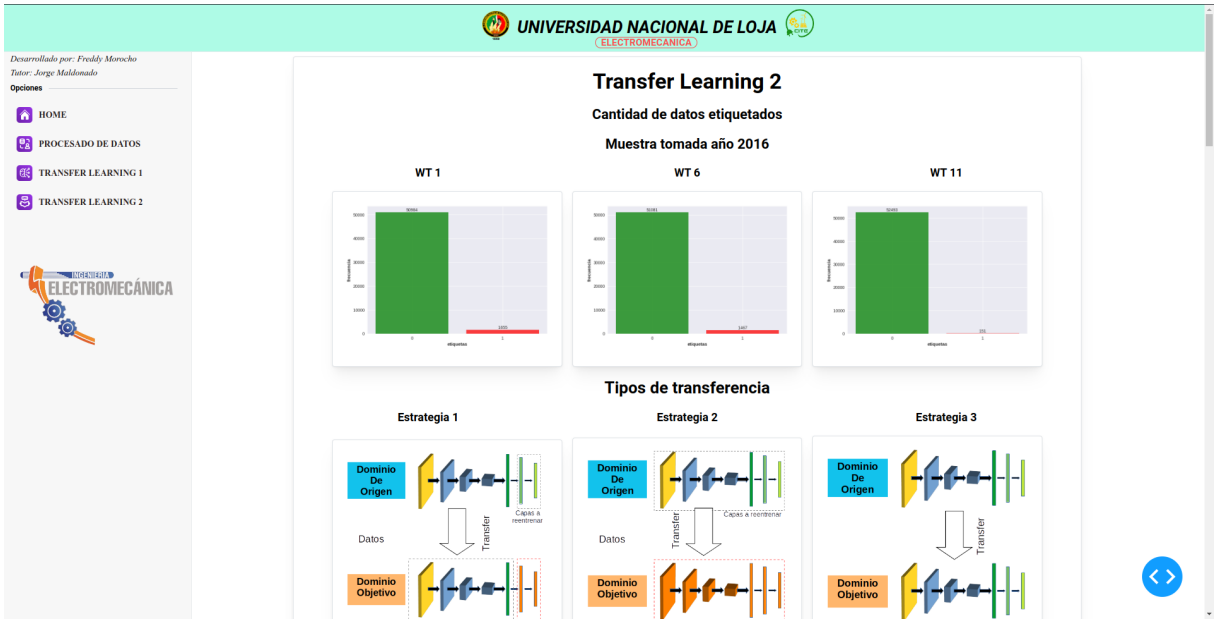


Figura 48. Ventana TL2 del dashboard.

7. Discusión

Esta investigación destaca el impacto positivo de la implementación de Transfer Learning en la detección de fallos de WTs, subrayando los beneficios derivados de la utilización de un dashboard para la visualización de resultados y la toma de decisiones. Los resultados obtenidos no solo permitieron la optimización y mejora del modelo predictivo, sino que también condujeron a la obtención de métricas altamente competitivas. Estos logros representan un cumplimiento efectivo de los objetivos propuestos, evidenciando así el potencial y la eficacia de la aplicación del Transfer Learning en el contexto de la predicción de fallos en WTs, respaldado por una herramienta visual como el dashboard para una interpretación más profunda y una toma de decisiones fundamentada.

Completado el proceso de data mining, se identificó que de las setenta y dos variables iniciales, únicamente se seleccionaron veinte mediante el método VIF. Esta selección aseguró que las variables no presentaran multicolinealidad significativa entre sí, garantizando así una relación mínima entre ellas. El etiquetado de datos realizado contribuyó a disponer de información sobre el WT con mayor cantidad de fallos. Un enfoque similar usa Velandia-Cardenas et al. (2021), el cual emplea el análisis de componentes principales (PCA) en lugar del VIF para la selección de variables, junto con un algoritmo de random oversampling para abordar el desbalance de datos. No obstante, en nuestro caso, el uso de random oversampling fue descartado, ya que los resultados obtenidos con SMOTE fueron considerablemente superiores en el modelo LSTM.

La elección del WT5 como base se fundamentó en su característica distintiva de exhibir una mayor frecuencia de etiquetas con un valor de 1 maximizando la captura de información relevante, convirtiéndolo en un candidato ideal para diseñar el modelo base y transferir conocimientos a otros WTs. A diferencia de Simani et al. (2023), quien elige un WT como base y realiza transferencias secuenciales a otros WTs generando un bucle que analiza cada estrategia de TL en todos los WTs, en nuestro caso se optó por centrar el modelo base en un único WT para evaluar y comparar las estrategias de TL.

En la fase de diseño del algoritmo base, se observó que el modelo LSTM exhibió desempeño superior en comparación del RNN y GRU, esto indica una tendencia respaldada por numerosas investigaciones que prefieren esta arquitectura o híbridos de LSTM con otras redes. Por ejemplo Zhu y Zhang (2021), empleó una red LSTM para predecir siete tipos de fallos de un WT. A pesar de la disparidad en los valores de accuracy entre el modelo base y el de Zhu

y Zhang (2021), se destaca que esta diferencia se atribuye a metodologías y enfoques diversos, ya que evitamos el uso de ventanas temporales y generación de datos faltantes y que el enfoque principal que se da a las métricas se centra en el F1-score debido a que el TL2 utiliza datos desbalanceados.

Los resultados del TL1 reveló la capacidad de la estrategia 3 para generar alarmas antes que sean registrados por parte del sistema SCADA. Este hallazgo sugiere considerar un reetiquetado para mejorar el modelo base, una estrategia respaldada por Chen et al. (2021), quienes proponen un método similar centrado en datos de falla, logrando anticipar eventos hasta 6 horas y 1.5 días antes y a partir de estas alarmas reetiquetar los datos. En nuestro estudio, se logró anticipar eventos de fallo con un rango de aproximadamente dos horas a un máximo de 8 días, subrayando la utilidad de la anticipación en un intervalo de tiempo significativo; este resultado es altamente beneficioso, ya que es conocido que las fallas no se producen instantáneamente, sino que presentan un período de desarrollo.

Los resultados del TL2 indican una mejora sustancial en los tiempos de entrenamiento al pasar de minutos a segundos, es importante destacar que en este proceso se realizó un reetiquetado de los datos, considerando únicamente un tipo de paro y sin la aplicación de algoritmos de oversampling, a pesar de contar con datos altamente desbalanceados; la estrategia 2 superó significativamente a la estrategia 1, logrando un F1-score superior al 74 % en las muestras de los WTs, mientras que la estrategia 3 presentó resultados bajos, demostrando que las estrategias 1 y 2 aprovechan eficazmente el conocimiento previo del modelo base. Estos hallazgos concuerdan con Simani et al. (2023), quienes, a pesar de utilizar una metodología diferente para evaluar los resultados del TL, concluyen que las estrategias 1 y 2 son óptimas para conjuntos de datos diferentes al del modelo base, mientras que la estrategia 3 solo muestra convergencia efectiva en conjuntos de entrenamiento idénticos, un resultado que también se evidencia en el TL1.

Al realizar el reajuste de hiperparámetros en el TL2, conseguimos resultados altamente competitivos en comparación con otras investigaciones relacionadas, como es el caso de Velandia-Cardenas et al. (2021) que logró F1-score de 0.71 y 0.96 para datos desbalanceados y balanceados respectivamente; mientras que en nuestro trabajo, con el reajuste de hiperparámetros, alcanzamos un F1-score de 0.9, 0.863 y 0.78 para el WT1, WT6 y WT11 respectivamente, y en datos no balanceados; aplicando un TL con la estrategia 2 logramos que nuestro modelo base pueda alcanzar un F1-score de 0.896, muy por arriba del modelo base inicial.

La comparación con investigaciones previas, como la de Velandia-Cardenas et al. (2021),

resalta la competitividad y eficacia de nuestra metodología, subrayando la importancia del ajuste fino de los hiperparámetros para obtener un rendimiento óptimo en conjuntos de datos específicos y en la tarea de predicción de fallos en WTs.

La implementación del dashboard ha mejorado significativamente la visualización de los resultados obtenidos en los TL1 y TL2. La integración de *Dash* ha permitido una interacción más efectiva con los datos y las predicciones, lo cual está en línea con las conclusiones presentadas por Hossain (2019). La capacidad de iterar y explorar los resultados a través del dashboard ha proporcionado una herramienta valiosa para comprender de manera más profunda los patrones y las tendencias emergentes, facilitando así la interpretación y la toma de decisiones informadas basadas en los resultados obtenidos en las estrategias de TL implementadas. Este enfoque visual reforzado contribuye significativamente a la comprensión global de los resultados y fortalece la utilidad práctica de las estrategias en el TL1 y TL2 en el contexto de la predicción de fallos en centrales eólicos.

8. Conclusiones

Finalizado el presente trabajo de titulación, se destacan las siguientes conclusiones:

- Con ayuda de técnicas de Data Mining se logró reducir las variables del sistema SCADA de setenta y dos variables iniciales a veinte. Esta reducción significativa se realizó mediante el método de selección VIF para evitar la multicolinealidad en las variables. Este método de selección demostró ser el más efectivo en la reducción de variables, en comparación con otros como PCA y la correlación de Pearson. Además, el etiquetado de datos facilitó la obtención de información crucial sobre el WT con más fallos; en este sentido, los datos del WT5 se eligieron para entrenar el modelo base por su gran cantidad fallos. En resumen, el Data Mining se presenta como una herramienta esencial para el procesamiento y preparación de datos antes de entrenar un modelo de predicción; y la metodología utilizada en este trabajo no solo garantizó la eficacia del proceso, sino que también demostró su robustez en todos los algoritmos empleados.
- En la aplicación de Transfer Learning (TL) para la detección de fallos en aerogeneradores, hemos logrado resultados significativos. La elección del modelo LSTM como base demostró superioridad sobre otros como RNN y GRU, validando la decisión mediante comparaciones con otros estudios previos.

  La implementación de estrategias de TL, tanto en anticipación de fallos (TL1), como en el diseño de un modelo predictivo con conjuntos de datos pequeños y altamente desbalanceados (TL2), reveló resultados prometedores. La capacidad del modelo para anticipar eventos en un rango de dos horas a ocho días, refuerza la eficacia de la estrategia 3 en la detección temprana de fallos y su aplicación en dataset similares. Las estrategias 1 y 2 focalizadas a la adaptación de dataset pequeños y desbalanceados, destacan por su eficacia en conjuntos distintos al del modelo base, siendo la estrategia 2 más versátil que la estrategia 1 por su capacidad de reentrenar todas sus capas.

  En consecuencia, la aplicación de TL en la detección de fallos en aerogeneradores se revela como una técnica potente y eficaz, respaldada por la solidez de la metodología y los resultados en cada etapa de este trabajo de titulación. Estos resultados no solo contribuyen al avance del campo, sino que también sugieren que la implementación de TL puede coadyuvar en las tareas de operación y mantenimiento aerogeneradores.
- Se desarrolló un dashboard que constituye un valioso aporte para visualizar los resultados

en las estrategias del TL1, TL2, así como el proceso de Data Mining. Esta herramienta interactiva refuerza la utilidad práctica de las estrategias, proporcionando al usuario una herramienta para la toma de decisiones con base en la predicción de fallos del aerogenerador. La implementación no solo beneficia a parques eólicos en operación, sino que también se posiciona como un complemento a los sistemas SCADA, contribuyendo así a la mejora continua de las tareas de operación y mantenimiento de aerogeneradores.

9. Recomendaciones

Referentes a las conclusiones presentadas, se proponen las siguientes recomendaciones para mejorar la investigación y posteriores trabajos.

- Es crucial destacar la mejora en la capacidad de anticipación que ofrece el modelo, sin embargo, es esencial abordar la posibilidad de falsos positivos. Este fenómeno puede manifestarse en cualquier sistema predictivo y, por ende, se recomienda una evaluación cautelosa al interpretar los resultados.

  La anticipación temprana de eventos es valiosa, pero su utilidad puede verse comprometida si no se gestiona adecuadamente la incidencia de falsas alarmas. Para garantizar la fiabilidad del modelo en situaciones operativas, se sugiere implementar estrategias adicionales para minimizar la ocurrencia de falsos positivos. Esto podría incluir ajustes en los umbrales de detección, consideración de características específicas de cada aerogenerador, y evaluación constante del desempeño del modelo en condiciones operativas reales.
- Se propone investigar la viabilidad de implementar el modelo en tiempo real en colaboración con el sistema SCADA. Esta integración proporcionaría una respuesta instantánea y permitiría acciones inmediatas ante posibles fallos, mejorando así la eficiencia operativa y la seguridad de los parques eólicos.
- Una línea de investigación futura sugerida es la expansión y mejora del modelo actual para abordar todos los tipos de fallos en lugar de limitarse a un enfoque binario. La implementación de un modelo multiclases y junto a la aplicación continua del TL permitirá una detección más precisa y versátil de fallos en aerogeneradores. Implementar TL individualizado para cada aerogenerador podría permitir mejoras constantes y adaptaciones a cambios futuros, consolidando finalmente una red única, la cual pueda ser incorporada a arquitecturas como la Mixter of Experts (MoE) para optimizar aún más la precisión y la generalización del sistema.

Estas recomendaciones pretenden abrir nuevas perspectivas para la investigación, buscando no solo la expansión y perfeccionamiento del modelo, sino también su aplicación práctica y continua mejora en entornos operativos reales.

10. Bibliografía

Abirami, S., & Chitra, P. (2020). Energy-efficient edge based real-time healthcare support system (P. Raj & P. Evangeline, Eds.). *117*(1), 339-368. https://doi.org/https://doi.org/10.1016/bs.adcom.2019.09.007

Alberto. (2013). *¿Qué es la Energía Eólica?* (Inf. téc.). CELECP. https://www.celec.gob.ec/gensur/index.php/contacto/direccion/2-uncategorised/47-que-es-laenergia-eolica

Albornoz, C. (2014). Diseño de interfaz gráfica de usuario. http://sedici.unlp.edu.ar/bitstream/handle/10915/41578/Documento_completo.pdf?sequence=1&isAllowed=y

Alonso, R. (2020, abril). IA, Machine Learning y Deep Learning, ¿cuál es la diferencia? https://hardzone.es/tutoriales/rendimiento/diferencias-ia-deep-machine-learning/

Al-Rahman, N. (2021, febrero). Undersampling and oversampling: An old and a new approach. https://medium.com/analytics-vidhya/undersampling-and-oversampling-an-old-and-a-new-approach-4f984a0e8392

Ayala, M., Maldonado, J., Paccha, E., & Riba, C. (2017). Wind Power Resource Assessment in Complex Terrain: Villonaco Case-study Using Computational Fluid Dynamics Analysis [3rd International Conference on Energy and Environment Research, ICEER 2016, 7-11 September 2016, Barcelona, Spain]. *Energy Procedia*, *107*, 41-48. https://doi.org/https://doi.org/10.1016/j.egypro.2016.12.127

Bello, E. (2021). *¿Qué es el minado de Datos o Data Mininig? Técnicas y pasos a seguir* (inf. téc.). https://www.iebschool.com/blog/data-mining-mineria-datos-big-data/

Blanco, J. I. (2023, abril). "¿Por qué la normalización es clave e importante en Machine Learning y Ciencia de Datos? https://jorgeiblanco.medium.com/por-qu%C3%A9-la-normalizaci%C3%B3n-es-clave-e-importante-en-machine-learning-y-ciencia-de-datos-4595f15d5be0

Brownlee, J. (2020). *Imbalanced Classification with Python: Better Metrics, Balance Skewed Classes, Cost-Sensitive Learning*. Machine Learning Mastery. https://books.google.com.ec/books?id=jaXJDwAAQBAJ

Byrnes, J. (2007). *Cognitive Development and Learning in Instructional Contexts* (3.). Pearson.

Chatterjeea, J., & Dethlefs, N. (2022). Scientometric Review of Artificial Intelligence for Operations Maintenance of Wind Turbines: The Past, Present and Future. https://www.researchgate.net/publication/359756812_Scientometric_Review_of_Artificial_Intelligence_for_Operations_Maintenance_of_Wind_Turbines_The_Past_Present_and_Future

Chen, W., Qiu, Y., Feng, Y., Li, Y., & Kusiak, A. (2021). Diagnosis of wind turbine faults with transfer learning algorithms. *Renewable Energy*, *163*, 2053-2067. https://doi.org/https://doi.org/10.1016/j.renene.2020.10.121

García, U. (2019, junio). Introducción a las Redes Neuronales. https://futurelab.mx/redes%20neuronales/inteligencia%20artificial/2019/06/25/intro-a-redes-neuronales-pt-1/

Gil, E. (2020, agosto). Métodos de Selección de Variables: El Factor de Inflación de la Varianza— profesorDATA.com. https://profesordata.com/2020/08/22/metodos-de-seleccion-de-variables-el-factor-de-inflacion-de-la-varianza/

GOLDWIND. (2011). *GOLDWIND 1.5 MW Technical Description*. Goldwind Science y Technology Co. LTD. www.goldwindglobal.com

Gonzalez, L. (2023). *Sobreajuste y Subajuste en Machine Learning*. https://aprendeia.com/sobreajuste-y-subajuste-en-machine-learning/

Gujarati, D., & Porter, D. (2009). *Econometría*. McGRAW-HILL.

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, *9*(8), 1735-1780. https://doi.org/10.1162/neco.1997.9.8.1735

Hoens, T. R., & Chawla, N. V. (2013). Imbalanced Datasets: From Sampling to Classifiers. En *Imbalanced Learning* (pp. 43-59). John Wiley Sons, Ltd. https://doi.org/https://doi.org/10.1002/9781118646106.ch3

Hossain, S. (2019). Visualization of Bioinformatics Data with Dash Bio, 126-133. https://doi.org/10.25080/Majora-7ddc1dd1-012

Instituto Nacional De Eficiencia Energética y Energías Renovables INER. (2014). *Análisis comparativo entre el estudio técnico previo y los datos reales de explotación del parque eólico Villonaco* (inf. téc.). https://www.geoenergia.gob.ec/wp-content/uploads/downloads/2019/09/analisis_comparativo_entre_los_datos_de_los_estudios_tecnicos.pdf

Karpathy, A. (2015). The Unreasonable Effectiveness of Recurrent Neural Networks. https://karpathy.github.io/2015/05/21/rnn-effectiveness/

Kelleher, J., & Tierney, B. (2018). *Data Science*. MIT Press. https://books.google.es/books?id=UlpVDwAAQBAJ

Ketkar, N., & Moolayil, J. (2021). *Deep Learning with Python: Learn Best Practices of Deep Learning Models with PyTorch*. Apress. https://doi.org/https://doi.org/10.1007/978-1-4842-5364-9

Lahura, E. (2003). El coeficiente de correlación y correlaciones espúreas. https://gc.scalahed.com/recursos/files/r161r/w24082w/S7_01.pdf

Lemaître, G., Nogueira, F., & Aridas, C. K. (2017). Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning. *Journal of Machine Learning Research*, *18*(17), 1-5. http://jmlr.org/papers/v18/16-365.html

Li, G., Wang, C., Zhang, D., & Yang, G. (2021). An Improved Feature Selection Method Based on Random Forest Algorithm for Wind Turbine Condition Monitoring. *Sensors*, *21*(16). https://doi.org/10.3390/s21165654

Lopez, R. (2023). Ciencia de datos. *IAAR*. https://iaarbook.github.io/datascience/#:~:text=La%20Ciencia%20de%20Datos%20es,sea%20estructurados%20o%20no%20estructurados.

Mandeville, P. B. (2008). Tema 18:¿ Por qué se deben centrar las covariables en regresión lineal? *CIENCIA-UANL*, *11*(3), 13.

Montero, E. R. (2020). *Industria 4.0 Conceptos, tecnologías, habilidades y retos*. Pirámide. https://www.edicionespiramide.es/libro.php?id=6219009

Olarte, C., Botero, A., & Benhur, A. (2010). Importancia del mantenimiento industrial dentro de los procesos de producción. *Scientia Et Technica*. https://www.redalyc.org/articulo.oa?id=84917316066

Ortiz, D. (2022). *¿Qué es un dashboard y para qué se usa?* https://www.cyberclick.es/numerical-blog/que-es-un-dashboard

Pan, S. J., & Yang, Q. (2010). A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, *22*(10), 1345-1359. https://doi.org/10.1109/TKDE.2009.191

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., … Chintala, S. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. En *Advances in Neural Information Processing Systems 32* (pp. 8024-8035). Curran Associates, Inc. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D.,

Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*, 2825-2830.

Pineda, C. (2021). *Aprendiza automático y profundo en python*. Ediciones de la U.

Plotly Technologies Inc. (2015). *Dash Python User Guide*. https://dash.plotly.com/

Politi, P. (2022). *Wind Turbine Condition Monitoring Through Artificial Neural Networks Using SCADA Data* [Tesis de maestría, Politecnico Di Torino]. https://webthesis.biblio.polito.it/25400/1/tesi.pdf

Proyecto LaTex. (2023). LaTeX – A document preparation system. *Latex-project.org*. https://www.latex-project.org/

Python Software Fundation. (2023, marzo). The Python Tutorial. *Python.org*. https://docs.python.org/3/tutorial/index.html

Ramírez, J. (2022). *Implementación de sistema SCADA para célula robótica de paletizado* [Tesis de maestría, Universidad Politécnica de Madrid]. https://oa.upm.es/71941/1/TFM_JORGE_ANDRES_RAMIREZ_PALADINES.pdf

Rodríguez, D. (2023, julio). Gráficos de correlación en Seaborn: Mapas de calor y gráficos de pares. https://www.analyticslane.com/2023/07/27/graficos-de-correlacion-en-seaborn-mapas-de-calor-y-graficos-de-pares/

Sexto, L. (2018). Tipos de mantenimiento:¿cuántos y cuáles son? *Electromagazine*, 40-46. http://www.mantenimientomundial.com/notas/SEXTO_Tipos-Mantenimiento.pdf

Simani, S., Farsoni, S., & Castaldi, P. (2023). Transfer Learning for Fault Detection with Application to Wind Turbine SCADA Data. *Journal of Energy and Power Technology*, *05*(01), 011. https://doi.org/10.21926/jept.2301011

Soto, C. (2017). *Central Eólica Villonaco Energías Renovables, Sustentables, y Sostenibles* (inf. téc.). Universidad Técnica Particular de Loja. http://eprints.rclis.org/32290/1/Art%C3%ADculo%20E%C3%B3licos..pdf

spotfire. (2024). What is data mining? https://www.spotfire.com/glossary/what-is-data-mining

Tartakovsky, S., Clark, S., & McCourt, M. (2017, agosto). https://developer.nvidia.com/blog/sigopt-deep-learning-hyperparameter-optimization/

TECH4CDM. (2009). *La Energía Eólica en Ecuador* (inf. téc.). https://biblioteca.olade.org/opac-tmpl/Documentos/cg00289.pdf

Torres, M., Hernández, J., Hernández, B., & Chávez, O. (2021). Impact of oversampling algorithms in the classification of guillain-barré syndrome main subtypes, 20-31. https://doi.org/https://doi.org/10.17163/ings.n25.2021.02

Vasconez, B. (2019). *Diseño y evaluación de una arquitectura para la red de comunicaciones que utiliza el sistema SCADA, para optimizar procesos y recursos en una empresa petrolera* [Tesis de maestría, Escuela Superior Politécnica de Chimborazo]. http://dspace.espoch.edu.ec/bitstream/123456789/10003/1/20T01155.pdf

Velandia-Cardenas, C., Vidal, Y., & Pozo, F. (2021). Wind Turbine Fault Detection Using Highly Imbalanced Real SCADA Data. *Energies*, *14*(6). https://doi.org/10.3390/en14061728

Zhu, L., & Zhang, X. (2021). Time Series Data-Driven Online Prognosis of Wind Turbine Faults in Presence of SCADA Data Loss. *IEEE Transactions on Sustainable Energy*, *12*(2), 1289-1300. https://doi.org/10.1109/TSTE.2020.3042800

11. Anexos

Anexo 1. Código dashboard

https://drive.google.com/drive/folders/1tZEeAaQosbyRMwEUH2lg7NItS0YQ9I5R?usp=sharing

Anexo 2. Manual de usuario

https://drive.google.com/file/d/1KD0tLECZqp_ryK8Shkq0Ntdu7_7MeLnO/view?usp=drive_link

Anexo 3. Certificado de traducción abstract

Lic. Talia Soledad Cuenca Calva

0979987189

cuencatalia102001@gmail.com

Loja-Ecuador

Loja, 7 de diciembre del 2024

La suscrita, Talia Soledad Cuenca Calva, **LICENCIADA EN PEDAGOGIA DEL IDIOMA INGLÉS** (número de registro SENESCYT:1008-2023-2797576), a petición de la parte interesada y en forma legal.

CERTIFICA:

Que la traducción del documento adjunto solicitado por el **Sr. Freddy Santiago Morocho Cuenca**, con cédula de ciudadanía **No. 1150233854**, cuyo tema de investigación se titula: **"Transfer learning para la detección de fallas en los aerogeneradores del parque Eólico Villonaco Loja-Ecuador."**, ha sido realizado y aprobado por mi persona, Talia Soledad Cuenca Calva, licenciada en pedagogía del idioma inglés.

El apartado del Abstract es una traducción textual del Resumen aprobado en español.

Particular que comunico en honor a la verdad para los fines académicos pertinentes, facultando al portador del presente documento, hacer el uso legal pertinente.

Firmado electrónicamente por:
TALIA SOLEDAD
CUENCA CALVA

Lic. Talia Soledad Cuenca Calva

English Teacher at STILL LANGUAGE CENTER