Pontificia Universidad
JAVERIANA
Cali

Facultad de Ingeniería y Ciencias

Ingeniería Biomédica

Informe Final de Trabajo de Grado

Desarrollo de una herramienta para la clasificación de movimientos de la mano asociados al síndrome del túnel carpiano mediante aprendizaje automático

Karen Nicolle Arango Valencia

Director
M.Sc. Juan Esteban Palacios Duarte

Codirector
M.Sc. Valentina Corchuelo Guzmán

1 de febrero de 2026

Santiago de Cali, 1 de febrero de 2026

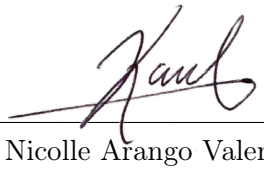
Señores
Pontificia Universidad Javeriana – Cali
Dra. María Constanza Pabón
Decano
Facultad de Ingeniería y Ciencias
Ciudad

Cordial Saludo.

Por medio de la presente me permito presentarle el Trabajo de Grado titulado "Desarrollo de una herramienta para la clasificación de movimientos de la mano asociados al síndrome del túnel carpiano mediante aprendizaje automático".

Espero que este trabajo reúna todos los requisitos académicos, cumpla el propósito para el cual fue creado y sirva de apoyo para futuros proyectos relacionados con la profesión.

Atentamente,



Karen Nicolle Arango Valencia

Santiago de Cali, 1 de febrero de 2026

Señores
Pontificia Universidad Javeriana – Cali
Dra. María Constanza Pabón
Decano
Facultad de Ingeniería y Ciencias
Ciudad

Cordial Saludo.

Certificamos que el presente Trabajo de Grado titulado "Desarrollo de una herramienta para la clasificación de movimientos de la mano asociados al síndrome del túnel carpiano mediante aprendizaje automático", realizado por Karen Nicolle Arango Valencia, estudiante de Ingeniería Biomédica, se encuentra terminado y puede ser presentado para su sustentación.

Atentamente,

M.Sc. Juan Esteban Palacios Duarte
Director Trabajo de Grado

M.Sc. Valentina Corchuelo Guzmán
Co-Director Trabajo de Grado

Agradecimientos

A la profesora Valentina Corchuelo y al profesor Juan Palacios, por su guía y orientación a lo largo de este proyecto. Su experiencia y dedicación fueron fundamentales para alcanzar los objetivos propuestos.

A mi familia, amigos y pareja, por su apoyo incondicional en cada etapa de este proceso, por su paciencia en los momentos difíciles y por compartir conmigo cada logro alcanzado.

A todos quienes de una u otra forma aportaron sus conocimientos y tiempo para hacer posible este trabajo, mi más sincero agradecimiento.

Glosario

Acrónimos y Abreviaturas

ML	Machine Learning conjunto de métodos que aprenden patrones a partir de datos mediante el uso de características definidas manualmente.
DL	Deep Learning rama del aprendizaje automático que emplea redes neuronales con múltiples capas para extraer automáticamente representaciones complejas de los datos.
sEMG	Electromiografía de superficie. Técnica no invasiva que registra la actividad eléctrica generada por los músculos mediante electrodos colocados sobre la piel.
CRISP-DM	Cross-Industry Standard Process for Data Mining. Metodología estándar utilizada en proyectos de minería y análisis de datos.
STC	Síndrome del Túnel Carpiano. Trastorno musculoesquelético causado por la compresión del nervio mediano en el túnel carpiano de la muñeca.
AHP	Analytic Hierarchy Process / Proceso Analítico Jerárquico. Método de decisión multicriterio que permite comparar de manera sistemática diferentes alternativas con base en múltiples criterios ponderados.
BCTQ	Boston Carpal Tunnel Questionnaire. Instrumento estandarizado de evaluación clínica utilizado para medir la severidad de los síntomas y el nivel de discapacidad funcional en pacientes con síndrome del túnel carpiano.
NCS	Nerve Conduction Study, es una prueba médica que mide la velocidad y la eficacia con la que las señales eléctricas viajan a través de los nervios.
CSA	Cross-Sectional Area. Área de Sección Transversal. Parámetro utilizado en estudios de imagen, especialmente con ultrasonido, para medir el área del nervio mediano a nivel del túnel carpiano.
AUC	Métrica de rendimiento para evaluar la eficacia de los modelos de clasificación binaria
IEEE	The Institute of Electrical and Electronic Engineers

Resumen

El síndrome del túnel carpiano (STC) es una neuropatía que se genera por la compresión del nervio mediano y compromete la función de la mano constituyendo una causa frecuente de ausentismo laboral. Su detección temprana es limitada por la falta de herramientas objetivas para evaluar patrones de movimiento asociados al riesgo biomecánico. En este proyecto se desarrolla una herramienta para clasificar movimientos de la mano vinculados al STC a partir de señales de sEMG y aprendizaje automático.

A partir de una revisión de repositorios públicos y empleando el Proceso Analítico Jerárquico (AHP), se seleccionó NinaPro DB2 (Ejercicio B), que incluye movimientos funcionales con gestos de riesgo como la flexión, extensión y desviaciones de la muñeca. Las señales provenientes de la base de datos se filtraron (20–450 Hz, notch 50 Hz), normalizaron mediante Z-score y segmentaron en 44 526 ventanas, con un desbalance de clases cercano a 1:4 (riesgo vs. seguro). Se entrenaron modelos clásicos basados en 144 características (Ensemble Subspace KNN y Random Forest) y modelos profundos sobre secuencias crudas (CNN-LSTM y BiLSTM), evaluando distintas estrategias de balanceo: sobremuestreo sintético en ML y data augmentation temporal y Focal Loss en DL.

Los resultados muestran que el desempeño depende fuertemente del tratamiento del desbalance. Con ADASYN, el modelo Random Forest alcanzó un AUC de 0.906, una sensibilidad del 86 % para la clase de riesgo y un F1-score de 0.83, siendo el clasificador más robusto. En los modelos profundos, la mejor configuración se obtuvo combinando únicamente data augmentation (jittering, scaling y time-warping), donde la CNN-LSTM alcanzó un AUC de 0.764, una sensibilidad cercana al 61 % y un F1-score de 0.62, mientras que el BiLSTM logró una sensibilidad del 65 % y un F1-score de 0.59. En conjunto, los modelos clásicos con sobremuestreo sintético capturan con mayor eficacia las diferencias entre movimientos seguros y de riesgo en bases con variabilidad temporal limitada.

El sistema se integró en un prototipo funcional con backend en Flask e interfaz web, que replica el preprocesamiento y permite realizar inferencia sobre nuevas señales sEMG. Los resultados demuestran la viabilidad de un enfoque no invasivo para monitorear patrones motores asociados al riesgo de STC y sientan las bases para herramientas de prevención accesibles en entornos ocupacionales.

Palabras Clave: aprendizaje automático, clasificación de movimientos, electromiografía de superficie, salud ocupacional, síndrome del túnel carpiano.

Abstract

Carpal tunnel syndrome (CTS) is a neuropathy caused by compression of the median nerve, impairing hand function and representing a frequent cause of work absenteeism. Its early detection is limited by the lack of objective tools to evaluate movement patterns associated with biomechanical risk. This project develops a tool to classify hand movements related to CTS using sEMG signals and machine learning.

Based on a review of public sEMG repositories and using the Analytic Hierarchy Process (AHP), the NinaPro DB2 dataset (Exercise B) was selected. This dataset includes functional movements, including risk-related gestures such as wrist flexion, extension, and radial/ulnar deviations. The signals were filtered (20–450 Hz, 50 Hz notch), normalized using Z-score, and segmented into 44,526 windows, with a class imbalance of approximately 1:4 (risk vs. safe). Classical models based on 144 handcrafted features (Ensemble Subspace KNN and Random Forest) and deep models operating on raw sequences (CNN-LSTM and BiLSTM) were trained, evaluating different balancing strategies: synthetic oversampling for ML and temporal data augmentation and Focal Loss for DL.

Results show that performance strongly depends on how the class imbalance is handled. With ADASYN, the Random Forest model achieved an AUC of 0.906, a sensitivity of 86 % for the risk class, and an F1-score of 0.83, making it the most robust classifier. In deep learning models, the best configuration was obtained using only temporal data augmentation (jittering, scaling, and time-warping), where the CNN-LSTM reached an AUC of 0.764, a sensitivity of about 61 %, and an F1-score of 0.62, while the BiLSTM achieved a sensitivity of 65 % and an F1-score of 0.59. Overall, classical models combined with synthetic oversampling captured differences between safe and risk movements more effectively, particularly in datasets with limited temporal variability.

The system was integrated into a functional prototype using a Flask backend and a web interface that replicates the preprocessing pipeline and enables inference on new sEMG signals. The results demonstrate the feasibility of a non-invasive approach for monitoring motor patterns associated with CTS risk and establish a foundation for accessible prevention tools in occupational settings.

Keywords: carpal tunnel syndrome, machine learning, movement classification, occupational health, surface electromyography.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

El síndrome del túnel carpiano (STC) es una de las neuropatías periféricas más frecuentes que afectan el miembro superior. En Colombia, se ha convertido en una enfermedad laboral muy común, especialmente en personas que desempeñan actividades repetitivas con las manos, como técnicos de laboratorio, personal de aseo y operarios industriales [1]. Este síndrome es causado por la compresión del nervio mediano en el túnel carpiano, lo que genera síntomas progresivos que incluyen entumecimiento, debilidad muscular y dolor.

El diagnóstico del STC se basa en exámenes clínicos como la electromiografía con agujas o los estudios de conducción nerviosa. Sin embargo, estos procedimientos suelen ser incómodos para el paciente y de difícil acceso en contextos con baja cobertura en salud [2, 3]. Por ello, existe la necesidad de desarrollar herramientas tecnológicas no invasivas que permitan identificar patrones predisponentes para el desarrollo del STC.

En este contexto, la electromiografía de superficie (sEMG) se presenta como una alternativa viable para capturar la actividad eléctrica muscular asociada a movimientos de la mano, lo cual combinado con técnicas de aprendizaje automático, permite construir modelos capaces de clasificar gestos y patrones de movimiento con alta precisión. Esto permitiría detectar aquellos movimientos que, por su frecuencia, incrementan el riesgo de desarrollar STC.

Este proyecto fue diseñado bajo la metodología CRISP-DM (Cross Industry Standard Process for Data Mining). Se plantea el desarrollo de una herramienta que integre el procesamiento de señales sEMG y algoritmos de aprendizaje automático, con el fin de identificar movimientos de la mano asociados a la sobrecarga biomecánica en el túnel carpiano.

El proyecto se enmarca en el contexto de la ingeniería biomédica aplicada a la salud ocupacional, buscando promover la detección temprana y la prevención de trastornos musculoesqueléticos mediante el uso de tecnologías accesibles. De esta forma, la investigación no solo aporta un avance técnico en el campo del procesamiento de señales biomédicas, sino también un impacto social al contribuir al bienestar de trabajadores expuestos a movimientos repetitivos y condiciones ergonómicas de riesgo.

Capítulo 2

Planteamiento del Problema

El síndrome del túnel carpiano (STC) es una de las neuropatías periféricas del miembro superior más comunes en la actualidad. Se estima que el 9.4 % de los trastornos musculoesqueléticos en las extremidades superiores afectan la muñeca y las manos, de los cuales el STC representa aproximadamente el 1.5 % [4]. De acuerdo con el Ministerio de Salud y Protección Social, en Colombia se consolida el síndrome del túnel carpiano como la primera causa de morbilidad profesional en el régimen contributivo. Dicha patología pasó de representar el 27 % de todos los diagnósticos en el año 2001, a representar el 32 % de los diagnósticos realizados durante el año 2004, presentando una tendencia continua al incremento [1].

El STC se debe a la compresión del nervio mediano dentro del túnel carpiano, lo cual provoca síntomas característicos como adormecimiento, dolor, pérdida de sensibilidad y debilidad muscular [5]. La compresión del nervio mediano ocurre en un estrecho canal anatómico delimitado por los huesos carpianos y el ligamento transverso del carpo, conocido como túnel carpiano.

Diversos estudios se han centrado en identificar factores de riesgo asociados al desarrollo del STC, encontrando que los más frecuentes incluyen el sexo femenino, la edad avanzada (principalmente entre 40 y 60 años) [6], y condiciones médicas preexistentes como diabetes y artritis [7, 8]. Adicionalmente, existen causas biomecánicas directamente relacionadas con actividades repetitivas que involucran movimientos específicos como la flexión y extensión constante de la muñeca, pinzas digitales prolongadas, agarres forzados o exposición continua a vibraciones [5, 9, 10].

Estos factores biomecánicos que incrementan progresivamente la presión interna del túnel carpiano, afectando el nervio mediano y reduciendo su función normal, son comunes en profesiones que implican tareas repetitivas manuales, tales como empleados de oficina, técnicos de laboratorio y neurocirujanos [11, 12]. Es así que estos profesionales ven disminuida su capacidad para realizar movimientos precisos como sujetar objetos pequeños, escribir o ejecutar tareas que requieren motricidad fina, generando una reducción importante en su calidad de vida, desempeño laboral, e incluso, en su bienestar emocional [13, 14].

Sin embargo, no solamente los pacientes se encuentran afectados, pues el STC representa una carga económica considerable para las empresas y los sistemas de salud debido a los costos asociados con tratamientos médicos, ausencias laborales y pérdida de productividad. En el 2014 el costo del tratamiento del STC en EPS está reportado entre $2 y $4 millones de pesos incluyendo posibles complicaciones y lesiones laborales que pueden representar costos totales de hasta $12000 millones

de pesos al año [15].

Desde una perspectiva clínica, el diagnóstico del STC tradicionalmente se basa en evaluaciones clínicas y pruebas especializadas como los estudios de conducción nerviosa, electromiografía y ecografía. El tratamiento varía dependiendo de la severidad del caso, oscilando desde terapia física y medidas conservadoras hasta la necesidad de intervención quirúrgica en etapas avanzadas [16].

A pesar de que existen métodos clínicos efectivos para diagnosticar el STC, la identificación temprana y objetiva de alteraciones motoras leves aún presenta limitaciones significativas. Esto se debe, principalmente, a la falta de herramientas tecnológicas automatizadas capaces de detectar con precisión los movimientos y patrones motores asociados con el desarrollo del síndrome. Por esta razón, el presente proyecto plantea el desarrollo de una herramienta no invasiva que permita identificar dichos movimientos, contribuyendo así a la prevención del STC.

Capítulo 3

Justificación

El STC es una enfermedad frecuente y costosa que afecta la función del nervio mediano y que se presenta con mayor intensidad en adultos en edad laboral. Aunque su prevalencia general se estima entre el 1–5 %, puede aumentar hasta 14.5 % en ciertos grupos ocupacionales [4]. Además, los estudios muestran diferencias claras según la exposición laboral: en trabajadores de aseo, las mujeres tienen 9 veces más probabilidad de desarrollar STC (con un 89 % de los casos atribuibles a la profesión), mientras que los hombres alcanzan un OR de 143, donde el 99 % de los casos se explica por la actividad desempeñada [14]. Esto deja ver la magnitud del problema, tanto por sus implicaciones clínicas como por los costos que representa para el sistema de salud y las empresas, tal como lo ha señalado el Ministerio de la Protección Social [1].

Más allá de los aspectos estadísticos, el STC afecta de manera directa la calidad de vida de los pacientes. Los síntomas (entumecimiento, dolor nocturno, debilidad y pérdida de precisión) dificultan actividades cotidianas y generan repercusiones emocionales importantes [14]. A esto se suma que los métodos diagnósticos actuales, como la electromiografía con aguja y los estudios de conducción nerviosa, pueden resultar invasivos, incómodos y de difícil acceso en regiones con baja cobertura en salud [17]. Por este motivo, resulta pertinente avanzar hacia herramientas no invasivas, portátiles y accesibles que faciliten la identificación temprana de factores predisponentes.

En el plano teórico, distintos estudios han demostrado el potencial del aprendizaje automático aplicado a señales biomédicas. Existen sistemas capaces de clasificar gestos de la mano con precisiones de hasta 98.70 % utilizando señales de sEMG [18]. De manera similar, modelos enfocados en la predicción del STC a partir de mediciones antropométricas y de fuerza han reportado precisiones cercanas al 89.47 % [19]. Estos resultados consolidan la idea de que es posible identificar patrones de movimiento asociados al riesgo antes de que la neuropatía avance.

Metodológicamente, este proyecto se centra en la elaboración de una herramienta móvil que permita clasificar movimientos de la mano asociados al STC mediante señales de sEMG y aprendizaje automático. El proyecto se desarrolla bajo la estructura de la metodología CRISP-DM, que permite organizar el proceso de manera clara y basada en datos. El uso de bases de datos públicas para el entrenamiento inicial de los modelos proporciona una base sólida y facilita la comparación con estudios previos.

La herramienta planteada tiene impacto en los ámbitos clínico, ocupacional y tecnológico. Puede contribuir a la detección temprana del STC sin procedimientos invasivos, apoyar el monitoreo

ergonómico de trabajadores expuestos a movimientos repetitivos y promover soluciones basadas en inteligencia artificial que sean accesibles, portátiles y útiles para la prevención de trastornos musculoesqueléticos.

Capítulo 4

Objetivos

4.1. Objetivo General

Desarrollar una herramienta para la identificación de movimientos de la mano relacionados con el síndrome del túnel carpiano mediante procesamiento de señales de electromiografía y aprendizaje automático.

4.2. Objetivos Específicos

- Gestionar una base de datos de señales EMG para la identificación de movimientos de la mano asociados al desarrollo del síndrome del túnel carpiano.
- Entrenar modelos de aprendizaje automático para la clasificación de movimientos de la mano a partir de señales de electromiografía.
- Evaluar el rendimiento de los modelos entrenados para la clasificación de señales EMG a partir de métricas de desempeño.
- Implementar una interfaz para la interacción entre los datos de señales de EMG y el modelo entrenado.

Capítulo 5

Marco de Referencia

5.1. Áreas Temáticas

- Carpal tunnel syndrome
- Machine learning
- Pattern recognition
- Data processing
- Biomedical signal processing
- Occupational health
- Motion detection

5.2. Marco Teórico

5.2.1. Fisiopatología del síndrome del túnel carpiano (STC)

El túnel carpiano es un espacio anatómico ubicado en la muñeca. Su límite anterior es el ligamento anular anterior del carpo también conocido como el ligamento carpiano transverso, su límite lateral está constituido por los huesos trapecio y escafoides, su límite medial por los huesos piramidal, pisciforme y ganchoso, y su límite posterior que corresponde a la superficie volar del carpo, que es cóncava y forma el llamado surco carpiano. Dentro del túnel se encuentran diez estructuras, cuatro corresponden a los tendones del flexor común superficial de los dedos, cuatro a los tendones del flexor común profundo de los dedos, una al tendón del flexor largo del pulgar y la última estructura corresponde al nervio mediano [2].

Como se aprecia en la figura 5.1 el nervio mediano es la estructura más superficial, una vez este entra en la palma de la mano se divide en ramas musculares que inervan los músculos tenares (abductor corto del pulgar, oponente del pulgar y la cabeza superficial del flexor corto del pulgar), así como los dos lumbricales laterales. Luego se divide en tres nervios palmares comunes que se ramifican en nervios digitales palmares propios, los cuales proporcionan sensibilidad a la superficie volar del dedo índice, dedo medio y la mitad radial del dedo anular [2].

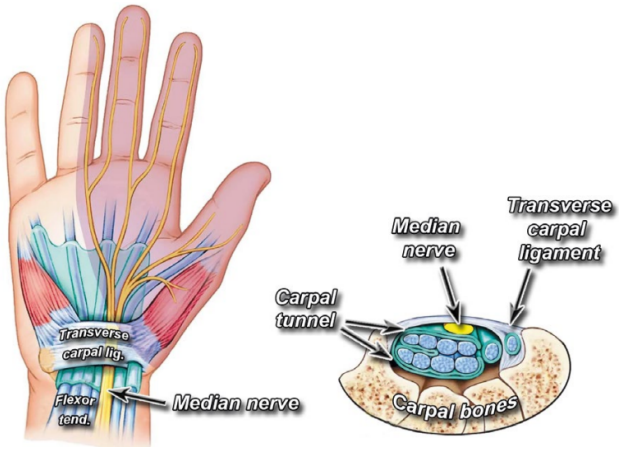


Figura 5.1: Anatomía del túnel carpiano

Debido al pequeño espacio que comparten las estructuras dentro del túnel carpiano, el nervio mediano es especialmente susceptible a la compresión, lo que deriva en el desarrollo de una de las neuropatías periféricas mas comunes conocida como síndrome del túnel carpiano (STC). Aunque el STC es una condición en gran parte idiopática, se han identificado múltiples factores de riesgo clínicos, ambientales, sociales y biomecánicos que influyen en su aparición.

Desde el punto de vista clínico, el STC está asociado con diversas condiciones médicas que incrementan el riesgo de compresión del nervio mediano en el túnel carpiano. Entre los factores extrínsecos que aumentan el volumen dentro del túnel carpiano se encuentran estados fisiológicos o patológicos que alteran el equilibrio de líquidos corporales, como el embarazo, la menopausia, la obesidad, la insuficiencia renal y el hipotiroidismo. Por otro lado, factores intrínsecos como tumores, quistes o masas dentro del túnel (producto de fracturas o artritis) también ocupan espacio adicional y predisponen a la compresión del nervio. Los factores neuropáticos, como la diabetes mellitus, el alcoholismo, la deficiencia o toxicidad vitamínica y la exposición a toxinas, son especialmente relevantes ya que afectan directamente la integridad del nervio mediano sin requerir un aumento de presión en el túnel [3, 7, 8].

Desde un enfoque social y demográfico, el STC muestra mayor prevalencia en en mujeres. El tamaño reducido del túnel carpiano, junto con un ligamento transverso más estrecho y una menor distensibilidad que los hombres (0.075 vs. 0.101 mm/N) podrían ser unas de las razones que expliquen la mayor prevalencia del STC en las mujeres [20]. Por otra parte, la permanencia prolongada en el entorno laboral también contribuyen al incremento en la incidencia del síndrome en personas entre los 40 y 64 años, particularmente en personas con condiciones médicas predisponentes[2, 3].

Por otro lado, la biomecánica del túnel carpiano es de vital importancia para determinar cómo se somete a estrés el nervio mediano. Los factores biomecánicos constituyen una de las causas más

documentadas del STC, entre ellos se incluyen tareas laborales que implican agarres con fuerza, movimientos repetitivos de flexión y extensión de la muñeca, así como exposición a herramientas vibrátiles. Estos movimientos aumentan la presión sobre el nervio mediano, se ha registrado que la presión normal dentro del túnel carpiano varía entre 2 mmHg y 10 mmHg. Sin embargo, el cambio en la posición de la muñeca puede provocar aumentos drásticos en la presión del fluido. Por ejemplo, la extensión de la muñeca puede elevar la presión a más de 10 veces su nivel inicial, mientras que la flexión puede generar un incremento de hasta 8 veces [3]. Como resultado, los movimientos repetitivos de la muñeca constituyen factores de riesgo importantes para la aparición del STC, en especial aquellos que generan presión sobre el nervio como lo serían la flexión, extensión y desviaciones (principalmente la desviación ulnar).

Por lo mencionado anteriormente la población mas afectada suelen ser personas expuestas a movimientos repetitivos y prolongados, como trabajadores manuales, personal de aseo, técnicos de laboratorio e incluso neurocirujanos [5, 11, 12]. Los síntomas que presentan las personas con STC incluyen hormigueo, entumecimiento, dolor, pérdida de fuerza de agarre y pinza, así como falta de coordinación de los dedos de la mano. La presentación clínica suele mostrar parestesias nocturnas que ocurren de forma intermitente y se vuelven más frecuentes durante la vigilia. Además, se presenta pérdida de sensibilidad en el pulgar, índice, medio y el lado radial del anular, lo cual coexiste con debilidad y, eventualmente, atrofia del músculo tenar a medida que la enfermedad progresa debido al daño axonal generalizado. Es importante destacar que el área tenar mantiene una sensibilidad normal, ya que está inervada por la rama sensitiva cutánea palmar, la cual no pasa por el túnel carpiano [2, 3].

El diagnóstico del STC se combina la evaluación clínica, pruebas provocativas, estudios electrodiagnósticos y técnicas de imagen. La historia clínica representa un componente esencial del proceso diagnóstico, ya que permite identificar las características y la distribución de los síntomas en la mano.

Las pruebas provocativas se utilizan para reproducir los síntomas del paciente mediante estímulos mecánicos sobre el nervio mediano. Entre las más empleadas se encuentran el signo de Tinel, que se considera positivo cuando el golpeteo repetido sobre el túnel carpiano provoca parestesias en la distribución del nervio; y la maniobra de Phalen, que consiste en flexionar la muñeca a 90 grados, se considera positiva si esta postura induce síntomas característicos del STC. Más recientemente, se ha introducido el "scratch collapse test", una prueba neurológica en la que se aplica un estímulo sobre la zona de compresión mientras el paciente realiza rotación externa de los hombros. La pérdida transitoria de resistencia muscular, que provoca un colapso del brazo, se interpreta como un resultado positivo.

Además de las pruebas físicas, existen instrumentos validados como el Boston Carpal Tunnel Questionnaire (BCTQ), el cual permite cuantificar de forma estandarizada los síntomas y el grado de discapacidad funcional del paciente. Este cuestionario, ampliamente utilizado a nivel internacional y traducido a múltiples idiomas, ha mostrado buena correlación con los hallazgos clínicos y los

resultados de conducción nerviosa.

Los estudios electrodiagnósticos son fundamentales para confirmar el diagnóstico, establecer la severidad y descartar patologías asociadas. Las pruebas de conducción nerviosa (NCS) permiten identificar la alteración específica en la conducción del nervio mediano a nivel de la muñeca, confirmando así la presencia de neuropatía mediana sin afectar otras regiones [2]. La electromiografía (EMG), por su parte, evalúa los cambios patológicos en los músculos inervados por el nervio mediano, en particular el abductor corto del pulgar, detectando signos de desnervación en casos avanzados.

En cuanto a los métodos de imagen, el ultrasonido puede usarse como una técnica no invasiva y confiable para la evaluación del STC. Permite observar signos característicos como el engrosamiento del nervio mediano, su aplanamiento dentro del túnel carpiano y cambios en el retináculo flexor. Uno de los parámetros más utilizados es el área de sección transversal (CSA) del nervio, medida a nivel del hueso pisiforme. Un CSA igual o superior a 9mm^2 ha mostrado una sensibilidad del 87.3 % y una especificidad del 83.3 % para el diagnóstico de STC [2].

El tratamiento del STC se selecciona en función de la gravedad del cuadro clínico. En casos leves a moderados, el abordaje inicial es conservador o no quirúrgico, incluyendo el uso de férulas en posición neutra durante la noche por al menos seis semanas, infiltraciones locales con corticosteroides, medicamentos orales como la prednisolona, fisioterapia, ejercicios de deslizamiento neural, movilización carpiana y ultrasonido terapéutico [16]. Estas estrategias buscan reducir la inflamación del tenosinovio, aliviar la compresión del nervio mediano y mejorar la función, particularmente cuando los síntomas tienen potencial de reversibilidad. Si no hay mejoría clínica, si el déficit motor o sensitivo progresa, o si los estudios electrodiagnósticos evidencian daño nervioso severo se suele recomendar intervención quirúrgica.

El tratamiento quirúrgico consiste en la liberación del túnel carpiano, mediante el corte longitudinal del ligamento transverso del carpo para descomprimir el nervio mediano. Esta intervención puede realizarse mediante técnica abierta o endoscópica, ambas han demostrado resultados funcionales comparables a largo plazo. La cirugía abierta (OCTR) es la más utilizada por su efectividad, bajo índice de complicaciones y aplicabilidad en casos complejos, como lesiones ocupantes de espacio, deformidades óseas o cirugías previas fallidas. Por otro lado, la liberación endoscópica, aunque menos invasiva, tiene un costo mayor y un riesgo ligeramente superior de daño nervioso transitorio. La elección de la técnica depende del contexto anatómico y clínico del paciente. El tiempo de recuperación varía según la severidad de la compresión nerviosa, e incluye rehabilitación mediante fisioterapia y uso temporal de férulas. En general, la liberación quirúrgica del túnel carpiano representa una opción efectiva y definitiva cuando el manejo conservador no es suficiente para revertir la sintomatología ni detener la progresión del daño neurológico [2, 3].

5.2.2. Señales de electromiografía (EMG) y procesamiento de señales biomédicas

La electromiografia (EMG) es un proceso mediante el cual se mide la señal eléctrica producida por los músculos usando electrodos, ya sea de forma invasiva, con electrodos de aguja (nEMG) o no invasiva, con electrodos de superficie (sEMG); a nivel de una sola fibra muscular, una unidad motora individual, o de todo el músculo [21].

Para la adquisición de la señal se pueden utilizar electrodos monopolares o bipolares, y en algunos casos, una combinación de electrodos intramusculares y de superficie. Para registros en superficie, se utilizan los llamados multi-electrodos, colocados sobre una alfombrilla de silicona, en disposición lineal (tiras) o matricial (rejillas). Para reducir la interferencia de la señal, puede aplicarse una conexión a tierra mediante un electrodo de referencia ubicado lejos del área de registro (similar al ECG) [21].

Las bioseñales son la columna vertebral de las ciencias biomédicas, ya que actúan como portadoras fundamentales de información que permite la comprensión de diversas patologías e incentiva la creación de nuevas tecnologías. En esencia, las señales transmiten mensajes mediante patrones, amplitudes y frecuencias que varían en el tiempo.

Por lo anterior bioseñales como el EMG han sido ampliamente utilizadas en el ámbito del aprendizaje profundo. Como vemos adquirir y procesar estas señales con precisión es fundamental, para que los algoritmos aprendan las relaciones entre las distintas características [22]. Sin embargo, no toda la informacion recopilada al momento de la adquisición de las señales corresponde a la señal que se desea adquirir. Las frecuencias de la señal EMG varían entre 0.01 Hz y 10 kHz, dependiendo del tipo de examen (invasivo o no invasivo) [21]. Sin embargo, las frecuencias más útiles y relevantes suelen encontrarse en el rango de 20 a 500 Hz, como se ve en la Fig.5.2 de Martinek, el resto de frecuencias, dependiendo de sus características pueden clasificarse como algún tipo de ruido.

La señal de electromiografía (EMG), particularmente en su modalidad de superficie (sEMG), es altamente susceptible a la contaminación por diferentes tipos de ruido y artefactos. Esta contaminación puede alterar tanto las características en el dominio del tiempo como del espectro de frecuencia, comprometiendo análisis como el diagnóstico de patologías, la estimación de fatiga y el control de prótesis.

Los principales contaminantes de una señal EMG se agrupan en tres categorías de acuerdo con Boyer [23]:

Ruido de línea base (Baseline Noise), también denominado ruido inherente, incluye el ruido térmico generado por los sistemas de amplificación (ruido blanco gaussiano) y el ruido en la interfaz piel-electrodo. Este último se comporta como un filtro pasa-bajas, atenuando las componentes útiles

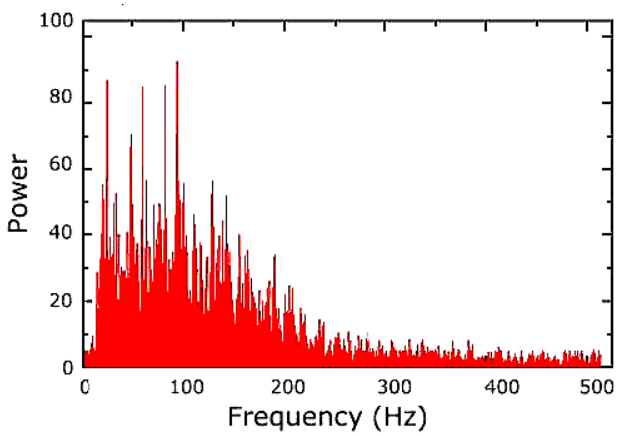


Figura 5.2: Ejemplo del espectro en frecuencia de una señal de sEMG

del EMG (>20 Hz) y conservando componentes indeseadas (<20 Hz), lo que disminuye la relación señal/ruido, el cual es un indicador que cuantifica cuán fuerte es una señal útil en comparación con el ruido que la acompaña.

Ruidos de interferencia, incluyen señales periódicas externas al EMG que contaminan el registro. Las más relevantes son:

- Señal electrocardiográfica (ECG), con contenido espectral hasta los 100 Hz.
- Crosstalk o señal EMG de músculos vecinos.
- Ruido electromagnético ambiental, siendo la interferencia de la red eléctrica la más común. Esta aparece como componentes en 50 o 60 Hz (según la región geográfica).

Artefactos de movimiento, que son perturbaciones no periódicas generadas por el movimiento del cuerpo o de los cables durante el registro. Alteran la impedancia de la interfaz piel-electrodo y se presentan típicamente en el rango de 0–20 Hz.

La mitigación de contaminantes en EMG se basa en una combinación de técnicas clásicas y modernas, enfocadas en preservar al máximo la integridad de la señal original, entre las mencionadas por Boyer y Martinek estan [23, 21]:

Filtros digitales convencionales:

- Filtros pasa-bajas, que eliminan componentes por encima de 400–500 Hz, asociadas a ruido blanco.
- Filtros pasa-altas, que atenúan artefactos de movimiento. Frecuencias de corte entre 20 y 30 Hz son recomendadas, aunque valores menores pueden afectar análisis de fatiga.

- Filtros elimina-banda (notch), usados para remover el ruido de la red electrica en 50/60 Hz y sus armónicos. También se usan filtros tipo peine (comb filters) o bandas de rechazo de 20–40 Hz para ECG.

Métodos de sustracción temporal: Estos métodos estiman la forma de onda del contaminante (por ejemplo, red eléctrica o ECG) y la restan directamente:

- Sustracción por regresión, que modela la interferencia como una suma de senos y cosenos, ajustando sus parámetros mediante regresión lineal.
- Sustracción espectral adaptativa, que realiza estimaciones armónicas variables en el tiempo mediante análisis en ventanas.
- Plantillas ECG, que usa la detección del complejo QRS para construir una plantilla promedio del ECG y sustraerla del EMG.

Estimación adaptativa de la interferencia: Técnicas que estiman la señal de ruido directamente para luego sustraerla:

- Promedio/madiana móvil, útiles para artefactos de movimiento.
- Filtros adaptativos, que son algoritmos como Least Mean Squares (LMS) y Recursive Least Squares (RLS) ajustan filtros FIR para cancelar el ruido, incluso sin referencia directa.
- Canceladores no lineales, como una red neuronal, ANFIS o filtros en espacio de estados cuando la interferencia tiene naturaleza no lineal.

Métodos de descomposición y reconstrucción: Aplican transformadas al dominio frecuencia o tiempo-frecuencia, eliminan contaminantes y reconstruyen la señal limpia:

- Transformada de Fourier (FFT): Estima el espectro del ruido (obtenido en reposo) y lo sustrae de la señal durante contracción.
- Transformada wavelet (DWT, SWT, WPT): Decomposición multiresolución donde se aplican umbrales a los coeficientes. Puede utilizar funciones HAD o SOF.
- Descomposición Empírica (EMD, EEMD, CEEMD): Divide la señal en modos intrínsecos (IMFs) y aplica umbrales. Útil contra ruido blanco, PLI y deriva basal.
- Descomposición por Modo Variacional (VMD): Variante de EMD adaptativa que produce mejores resultados, especialmente en señales con bajo SNR.

Métodos híbridos: Se combinan estrategias para mejorar el desempeño:

- Wavelet + ICA: Descomposición wavelet seguida de separación ciega de fuentes.
- Wavelet + ANC: Uso de WT para estimar la interferencia como entrada del cancelador adaptativo.
- Wavelet + Filtro de Wiener: Se construye una señal de referencia filtrada con WT y se aplica un Wiener adaptado.

La selección del método óptimo de eliminación de ruido en EMG depende del tipo de contaminante, la aplicación y la necesidad de preservar las características de la señal. Las técnicas modernas basadas en descomposición y métodos adaptativos permiten una mitigación más selectiva y eficiente, pero requieren un análisis más profundo de sus parámetros y efectos sobre la señal útil, por lo que los métodos de filtrado tradicional siguen siendo utilizados.

Con el aumento de las aplicaciones de las señales de EMG en el contexto del ML, se ha visto la necesidad de expandir significativamente la disponibilidad de grandes conjuntos de datos. La transición hacia el "big data" en EMG ha sido posible por el incremento en las tecnologías comerciales de adquisición, almacenamiento y distribución de datos, lo que facilita el acceso a registros antes solo encontrados en laboratorios particulares [24].

Esta diversidad es clave para identificar parámetros y características generalizables. Algunos estudios han demostrado que parámetros óptimos como los umbrales para características como cero cruces (ZC) o cambios de pendiente (SSC) son altamente dependientes del sujeto y del dataset, reforzando la necesidad de entrenar modelos en múltiples bases de datos[24].

Entre los conjuntos de datos más relevantes se encuentra el repositorio NinaPro, considerado como la base de datos de referencia más extensa en la actualidad. Esta base contiene señales EMG de más de 117 sujetos sanos y 13 amputados, capturadas en hasta 61 movimientos distintos de mano y dedos, totalizando más de 48,000 ensayos. Incluye señales adquiridas con distintos dispositivos como Otto Bock MyoBock, Delsys Trigno, Cometa Wave Plus y Myo armbands, e incorpora sensores de inercia, guantes de datos, inclinómetros, y otros canales fisiológicos. Aunque sus diferentes subconjuntos no son completamente compatibles entre sí, su contribución al campo ha sido sustancial al proporcionar una plataforma común de validación.

Otro avance notable es el uso de EMG de alta densidad (HD-sEMG), que emplea matrices de hasta 192 electrodos para capturar información espacial detallada del músculo. Estos datos permiten representar la actividad muscular como imágenes 2D, lo que abre la puerta a análisis mediante técnicas de procesamiento de imágenes y aprendizaje profundo, como redes convolucionales. Bases como CapgMyo y csl-hdemg ejemplifican este tipo de datasets.

También existen bases multimodales relevantes para tareas como reconocimiento de emociones, incluyendo señales EMG junto a EEG, ECG y GSR, como las bases DEAP, BioVid Emo DB y DECAF. Estas combinaciones han demostrado aumentar la robustez del reconocimiento en presencia de ruido o pérdida de señales.

A pesar de los avances, el campo aún carece de un protocolo estandarizado aceptado globalmente para adquisición EMG, lo cual dificulta la interoperabilidad entre conjuntos de datos [24].

5.2.3. Aprendizaje automático para clasificación de señales

En los últimos años, el interés por la Inteligencia Artificial (IA), y en particular por el Machine Learning (ML), ha crecido de forma exponencial. Esta disciplina se ha consolidado como una herramienta clave para el desarrollo e implementación de soluciones en múltiples sectores, incluyendo la industria, las ciencias sociales y especialmente el área biomédica. El Machine Learning se basa en la automatización del proceso de construcción de modelos analíticos, permitiendo que los sistemas informáticos aprendan a partir de los datos, sin intervención humana directa. Hoy en dia gracias al acceso a grandes volúmenes de información, estos sistemas pueden aprender a identifican patrones para generan predicciones o tomar decisiones fundamentadas [25].

De manera análoga al aprendizaje humano, donde intervienen la experiencia, los sentidos y estrategias cognitivas, los algoritmos de ML aprenden mediante la exposición a datos. Sin embargo, en el entorno informático este aprendizaje se formaliza mediante algoritmos especializados que dotan a las máquinas de autonomía y capacidad de adaptación. Para comprender mejor el funcionamiento de ML, es fundamental conocer sus principales enfoques de aprendizaje, de acuerdo con Costa [26] y como se observa en la Fig.5.3, existen varios tipos de machine learning, que pueden clasificarse en cuatro categorías principales [25]:

- Aprendizaje supervisado: el modelo es entrenado con datos etiquetados, lo que le permite aprender una relación entre los datos de entrada y las salidas esperadas.
- Aprendizaje no supervisado: se utiliza cuando los datos no están etiquetados, buscando estructuras o patrones ocultos sin una guía explícita.
- Aprendizaje semi-supervisado: es un enfoque intermedio que utiliza tanto datos etiquetados como datos no etiquetados para entrenar algoritmos. Normalmente, se utiliza una pequeña cantidad de datos etiquetados junto con una gran cantidad de datos no etiquetados.
- Aprendizaje por refuerzo: el sistema interactúa con un entorno y aprende a partir de las recompensas o penalizaciones recibidas por sus acciones, mejorando su rendimiento de forma iterativa.

Estos enfoques permiten el desarrollo de sistemas inteligentes adaptados a el tipo de problema que se desea abordar, lo que ha abierto nuevas posibilidades en campos como la salud y la bioinge-

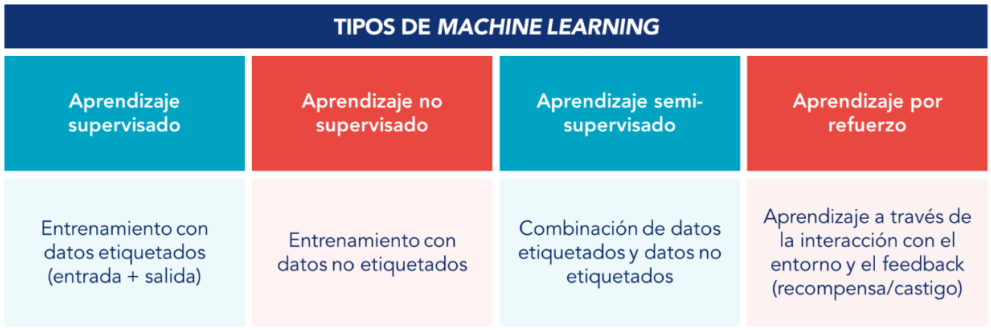


Figura 5.3: Principales tipos de modelos de machine learning

niería.

En particular, la ingeniería biomédica ha sido profundamente impactada por la integración de técnicas avanzadas de procesamiento de datos, entre las cuales el aprendizaje automático destaca por su capacidad de interpretar señales fisiológicas complejas. Un caso representativo es el análisis de señales de electromiografía (EMG), que se ha convertido en una técnica no invasiva ampliamente utilizada para registrar la actividad eléctrica generada por los músculos durante su contracción.

Específicamente, la señal EMG permite capturar patrones eléctricos producidos por los músculos, estas señales son fundamentales en las Interfaces Hombre–Máquina y pueden utilizarse en diversas aplicaciones como el reconocimiento de gestos de la mano. La singularidad de las señales EMG de cada persona hace necesario el desarrollo de sistemas capaces de aprender una gran cantidad y diversa de datos [22].

Como se muestra en la Fig.5.4, para cumplir con su tarea los sistemas de clasificación de ML suelen estructurarse en una arquitectura modular compuesta por varias etapas clave [27]:

- Colección de Datos: En esta etapa se adquieren los datos crudos, ya sea provenientes de sensores o de bases de tados con señales previamente adquiridas.
- Preprocesamiento de Datos: Consiste en aplicar técnicas para mejorar la calidad de las señales, incluyendo filtrado de ruido, normalización, segmentación en ventanas temporales y etiquetado.
- Exploración de Datos: Se analizan estadísticamente las características del conjunto de datos. Esta fase permite identificar patrones, relaciones entre variables, valores atípicos y posibles sesgos, proporcionando una base sólida para la toma de decisiones posteriores.
- Entrenamiento del Algoritmo: Aquí se selecciona un algoritmo de aprendizaje automático (como Random Forest, KNN o redes neuronales) y se entrena con el conjunto de datos procesados, permitiendo que el modelo aprenda a asociar patrones.

- Evaluación de los Algoritmos: Se valida el rendimiento del modelo usando métricas como precisión, sensibilidad, especificidad o el área bajo la curva (AUC). Para ello, se emplea un conjunto de prueba independiente o técnicas como validación cruzada. De acuerdo a los resultados se realizan cambios en los hiperparámetros del modelo y se repite el entrenamiento.

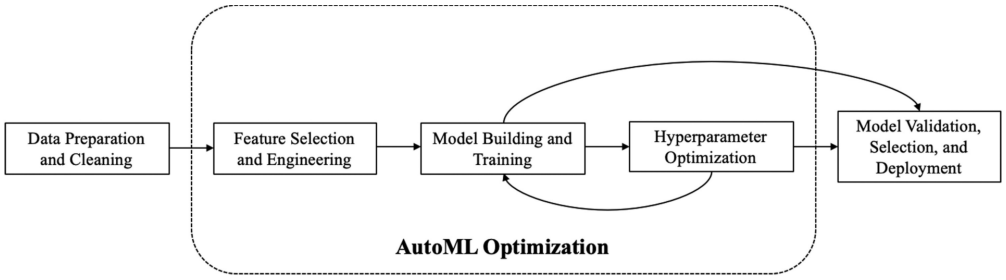


Figura 5.4: Capas de una arquitectura de machine learning

Una etapa crítica en el procesamiento y clasificacion de señales electromiográficas es la extracción de características, ya que permite transformar las señales crudas en representaciones numéricas que capturan información discriminativa relevante para tareas de clasificación o análisis clínico. De acuerdo con C.L.Kok, las características que se utilizan actualmente en la clasificación de señales EMG incluyen las siguientes [22]:

El **Valor Absoluto Medio (MAV)** estima la magnitud promedio de una señal al calcular el promedio de los valores absolutos de sus muestras. Es útil para evaluar el nivel de contracción muscular. Se define como:

$$\text{MAV} = \frac{1}{L}\sum_{i=1}^{L}|x_i| \tag{5.1}$$

La **Longitud de Onda (WL)** representa la complejidad de la señal al medir la longitud acumulada de sus oscilaciones, sumando las diferencias absolutas entre muestras consecutivas:

$$\text{WL} = \sum_{i=2}^{L}|x_i - x_{i-1}| \tag{5.2}$$

El **Cruce por Cero (ZC)** contabiliza las veces que la señal cambia de signo, lo que se asocia con la frecuencia del contenido muscular. Se emplea un umbral T para evitar contar transiciones causadas por ruido:

$$\text{ZC} = \sum_{i=1}^{L-1} f(x_i), \quad \text{donde } f(x_i) = \begin{cases} 1, & \text{si } (x_i > 0 \land x_{i+1} < 0) \lor (x_i < 0 \land x_{i+1} > 0) \land |x_i - x_{i+1}| \geq T \\ 0, & \text{en otro caso} \end{cases} \tag{5.3}$$

El **Cambio de Signo de la Pendiente (SSC)** cuantifica los cambios de dirección en la pendiente de la señal, lo cual es útil para detectar transiciones dinámicas. Se define mediante:

$$\text{SSC} = \sum_{i=2}^{L-1} f(x_i), \quad \text{donde } f(x_i) = \begin{cases} 1, & \text{si } [(x_i > x_{i-1} \wedge x_i > x_{i+1}) \vee (x_i < x_{i-1} \wedge x_i < x_{i+1})] \\ & \wedge (|x_i - x_{i+1}| \geq T \vee |x_i - x_{i-1}| \geq T) \\ 0, & \text{en otro caso} \end{cases} \tag{5.4}$$

El **Valor Cuadrático Medio (RMS)** es una medida de la energía media de la señal. Se emplea ampliamente para cuantificar la intensidad muscular y se calcula como:

$$\text{RMS} = \sqrt{\frac{1}{L} \sum_{i=1}^{L} x_i^2} \tag{5.5}$$

La **Frecuencia Media (Mean Frequency, f_{mean})** representa el promedio ponderado de las componentes de frecuencia de la señal, donde la ponderación está dada por la densidad espectral de potencia (PSD). Indica el "centro de masa" del espectro en términos de energía y se calcula como:

$$f_{\text{mean}} = \frac{\sum_{k=1}^{N} f_k \, P(f_k)}{\sum_{k=1}^{N} P(f_k)} \tag{5.6}$$

donde $P(f_k) = |\text{FFT}(x)|^2$ es la densidad espectral de potencia de la señal x, y f_k representa la frecuencia correspondiente al k-ésimo componente del espectro.

La **Frecuencia Mediana (Median Frequency, f_{med})** es la frecuencia que divide el espectro de potencia en dos mitades iguales, de modo que la mitad inferior del espectro contiene la misma energía que la mitad superior. Se obtiene resolviendo:

$$\sum_{k=1}^{k_{\text{med}}} P(f_k) = \frac{1}{2} \sum_{k=1}^{N} P(f_k) \tag{5.7}$$

donde $P(f_k)$ es la densidad espectral de potencia y $f_{k_{\text{med}}}$ es la frecuencia mediana correspondiente.

La **Energía Wavelet por Nivel (E_j)** cuantifica la energía contenida en cada subbanda de frecuencia obtenida mediante la descomposición wavelet discreta (DWT). Esta característica permite capturar la distribución de energía de la señal en diferentes escalas temporales. Se define como:

$$E_j = \sum_{n=1}^{L_j} |c_{j,n}|^2 \tag{5.8}$$

donde $c_{j,n}$ son los coeficientes wavelet en el nivel j, y L_j es el número de coeficientes en dicho nivel.

El vector de características resultante incluye las energías correspondientes a todos los niveles de descomposición:

$$\mathbf{F}_{\text{wavelet}} = [E_1, E_2, \ldots, E_J] \tag{5.9}$$

Estas características son fundamentales para la representación cuantitativa de la actividad muscular, y su adecuada selección e implementación puede incrementar significativamente el rendimiento de los modelos de clasificación aplicados a señales EMG.

5.2.4. Clasificadores supervisados para la clasificación de señales de EMG

Para el caso especifico de clasificación de movimientos de la mano a partir de señales EMG, nos enfocamos en el aprendizaje supervisado, que utiliza datos etiquetados para entrenar modelos capaces de realizar tareas de clasificación. A continuación se describen tres de los clasificadores más empleados en el procesamiento de señales EMG [22, 25]:

K-Nearest Neighbors (KNN) es un algoritmo de clasificación supervisado y no paramétrico. Su principio se basa en asignar una clase a una nueva muestra según la mayoría de clases presentes entre sus k vecinos más cercanos, calculados típicamente mediante la distancia euclidiana o la similitud coseno negativa.

Visualmente, este algoritmo agrupa los datos en regiones de decisión definidas por la vecindad de cada punto. Es simple de implementar, pero puede ser sensible a la escala de los datos y al valor de k seleccionado.

Random Forest es un clasificador de tipo ensamblado ("ensemble") que construye múltiples árboles de decisión independientes sobre muestras aleatorias del conjunto de entrenamiento (y reduciendo la correlación entre árboles al seleccionar sub conjuntos aleatorios de características)[28]. Luego combina sus predicciones por mayoría (en clasificación) o por promedio (en regresión). Esta estrategia mejora la precisión y reduce la varianza respecto a un único árbol de decisión.

El proceso puede resumirse como:

- Selección de muestras de entrenamiento con remuestreo (bootstrap) para cada árbol.
- Para cada árbol, selección aleatoria de un subconjunto de características en cada división del nodo.
- Cada árbol crece hasta cierta profundidad o criterio de parada y luego aporta su predicción.
- En clasificación, la clase final es la que obtiene el mayor número de votos de los árboles; en regresión, se promedian las salidas.

Una formulación simplificada de la predicción para clasificación es:

$$\hat{y} = \text{mode}\{\, h_b(x)\}_{b=1}^{B} \tag{5.10}$$

donde $h_b(x)$ es la predicción del b-ésimo árbol y B el número total de árboles del bosque.

Máquinas de Vectores de Soporte (SVM) son clasificadores supervisados robustos frente a ruido o valores atípicos. Su objetivo es encontrar el hiperplano óptimo que separe los datos de diferentes clases con el mayor margen posible. Cuando los datos no son linealmente separables en el espacio original, el algoritmo utiliza una técnica conocida como *kernel trick* para proyectar los datos a un espacio de mayor dimensión donde sí es posible realizar una separación lineal efectiva.

Esta propiedad convierte a SVM en una opción poderosa para problemas de clasificación compleja, incluyendo señales EMG multivariadas. El modelo puede emplear núcleos lineales, polinomiales, o basados en funciones de base radial (RBF), entre otros.

Por ultimo la evaluación del desempeño de un modelo de aprendizaje automático es fundamental para determinar su efectividad y capacidad predictiva. En problemas de clasificación, las métricas más empleadas incluyen la precisión, la sensibilidad, la exactitud y la F-medida (F1-score). Estas métricas se calculan a partir de los resultados de las predicciones realizadas por el modelo, específicamente utilizando los valores de verdaderos positivos (TP), verdaderos negativos (TN), falsos positivos (FP) y falsos negativos (FN) [22].

La **precisión** cuantifica la proporción de predicciones positivas que fueron correctas. Es decir, mide la capacidad del modelo para evitar clasificar incorrectamente como positivas aquellas instancias que en realidad son negativas. Se define como:

$$\text{Precisión} = \frac{TP}{TP + FP} \tag{5.11}$$

La **sensibilidad** (sensitivity o recall) representa la proporción de casos positivos correctamente identificados por el modelo. Evalúa la capacidad del modelo para evitar omitir instancias positivas reales. Su expresión matemática es:

$$\text{Sensibilidad} = \frac{TP}{TP + FN} \tag{5.12}$$

La **exactitud** (accuracy) calcula la proporción de predicciones correctas (positivas y negativas) sobre el total de muestras evaluadas. Si bien es útil como medida general del desempeño, puede resultar engañosa en conjuntos de datos desbalanceados. Se calcula mediante:

$$\text{Exactitud} = \frac{TP + TN}{TP + TN + FP + FN} \tag{5.13}$$

La **F-Medida (F1-Score)** es una métrica que combina la precisión y la sensibilidad en una sola expresión mediante su media armónica. Es especialmente útil cuando existe un desequilibrio entre clases, ya que penaliza los modelos que favorecen una métrica a costa de la otra. Se define como:

$$\text{F1-Score} = 2 \cdot \frac{\text{Precisión} \cdot \text{Sensibilidad}}{\text{Precisión} + \text{Sensibilidad}} \tag{5.14}$$

Con estas metricas se puede tener una visión integral del comportamiento del modelo, permitiendo identificar fortalezas y debilidades en escenarios diversos.

5.2.5. Modelos de deep learning

El deep learning (DL) es un subcampo del machine learning que emplea redes neuronales profundas para aprender representaciones jerárquicas directamente a partir de grandes volúmenes de datos y tomar decisiones con mínima intervención manual. Estas arquitecturas se inspiran en la organización del cerebro y pueden entrenarse en esquemas supervisados, semi supervisados o no supervisados [29].

Un modelo profundo está compuesto por capas de neuronas artificiales (capas de entrada, múltiples capas ocultas y capas de salida) conectadas mediante pesos y sesgos, como se aprecia en la imagen 5.5. La profundidad, es decir, el mayor número de capas, permite extraer patrones más complejos. El aprendizaje se realiza de manera iterativa ajustando los pesos mediante el algoritmo de backpropagation para minimizar una función de pérdida.

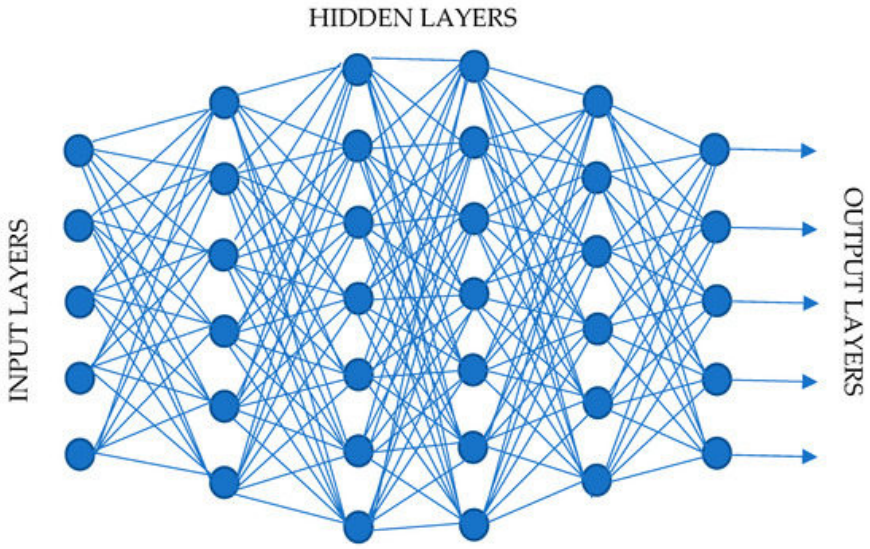


Figura 5.5: Representación de las capas de una red neuronal

Diferencias con métodos tradicionales de ML A diferencia de los clasificadores convencionales (como KNN o Random Forest), que dependen de una etapa explícita de extracción de características, los modelos profundos realizan aprendizaje de representaciones de extremo a extremo, optimizando simultáneamente la transformación de los datos y la clasificación. Por lo tanto, esto reduce el sesgo de diseño de características y permite transferir conocimiento entre dominios, conocido como transfer learning, aunque exige recursos computacionales significativos y medidas para evitar sobreajuste o pérdida de interpretabilidad [29].

Principales arquitecturas y usos. La literatura presenta diversas familias de modelos de DL, cada una diseñada para capturar tipos específicos de patrones y dependencias:

- **Redes Neuronales Artificiales (ANN / MLP):** Son la base del aprendizaje profundo. Están formadas por capas totalmente conectadas y se utilizan en problemas de regresión, clasificación tabular y como bloques en arquitecturas más complejas.
- **Redes Neuronales Convolucionales (CNN):** Introducen filtros convolucionales para detectar patrones espaciales y son ampliamente utilizadas en el análisis de imágenes médicas, electrocardiogramas y señales electromiográficas (EMG).
- **Redes Recurrentes (RNN), LSTM y GRU:** Modelan dependencias temporales, lo que las hace adecuadas para series temporales fisiológicas, texto clínico o datos longitudinales.
- **Autoencoders (AE):** Redes de tipo *encoder–decoder* que aprenden representaciones latentes para reducción de dimensionalidad o eliminación de ruido. Los **Denoising Autoencoders** se entrenan para reconstruir entradas limpias a partir de versiones con ruido, aumentando la robustez del modelo.
- **Variational Autoencoders (VAE):** Extienden los AE clásicos mediante un marco probabilístico que permite generar nuevas muestras, útiles para detección de anomalías y síntesis de datos biomédicos.

El DL ha demostrado tener gran impacto en múltiples áreas de la ingeniería biomédica, como en: el análisis de imágenes médicas, bioinformática, predicción de enfermedades y procesamiento de señales fisiológicas (EMG, EEG, ECG y EOG) [24].

En tareas de clasificación de señales electromiográficas, las etapas previas de normalización y segmentación son esenciales. Debe evaluarse si un pipeline clásico con extracción de características RMS, MAV, WL, ZC, es suficiente, o si un modelo profundo puede aprender representaciones directamente de las señales crudas. Ante la escasez de datos, se recomienda el uso de aumento de datos y regularización, como dropout y penalización o early stopping. Además, la validación cruzada por sujeto, repetición o sesión es fundamental para estimar la capacidad de generalización del modelo [29].

5.3. Trabajos Relacionados

La electromiografía de superficie ha cobrado creciente relevancia en aplicaciones de interacción humano máquina (HMI), debido a su capacidad para identificar movimientos mediante el registro de la actividad eléctrica muscular. El uso de modelos de aprendizaje automático y de aprendizaje profundo, ha sido ampliamente explorado para tareas como la clasificación de gestos, lo que sienta un precedente directo para el desarrollo de sistemas de apoyo diagnóstico en afecciones como el síndrome del túnel carpiano.

Un análisis sistemático reciente destaca las tendencias y métodos emergentes en el reconocimiento de actividades humanas basadas en sEMG [30]. En dicho estudio se recopilan y comparan diversos enfoques para el preprocesamiento, la extracción de características y el uso de clasificadores tradicionales (SVM, k-NN, Random Forest) y redes neuronales profundas (CNN, LSTM), reportando altos niveles de precisión en tareas como reconocimiento de actividades de la vida diaria (ADL) y gestos de la mano. Además, se enfatiza la importancia del diseño experimental, incluyendo aspectos como la colocación de electrodos, número de canales y protocolos de captura, lo cual influye directamente en la generalización del modelo. Esto es especialmente relevante para tareas clínicas, donde se requiere robustez intersujeto.

El trabajo fundacional de Atzori describe la base de datos Ninapro (DB1–DB3), diseñada para el estudio de la relación entre sEMG, cinemática y dinámica de la mano y parámetros clínicos, con 67 sujetos sanos y 11 amputados y hasta 50 movimientos por protocolo. En su validación técnica muestran que, empleando características clásicas (RMS, estadísticas en tiempo, histogramas, mDWT) y clasificadores estándar (k-NN, SVM, Random Forest, LDA), el accuracy promedio para 50 gestos alcanza 75.32 % (DB1) y 75.27 % (DB2) con Random Forest y combinación de todas las características. En amputados (DB3) se reporta 46.27 % para 50 gestos con SVM y todas las características. El artículo resalta la necesidad de reportes balanceados y también el cuidado al dividir repeticiones entre entrenamiento y prueba para evitar sesgos [31].

Ding et al. proponen una arquitectura CNN con bloques paralelos multiescala y núcleos de mayor tamaño que los habituales para capturar propiedades específicas del sEMG. Evaluada en Ninapro DB2, la red logra 78.86 % de accuracy y supera métodos previos (p. ej., 76.1 % con CNN single frame). El enfoque evita extracción manual de características y muestra que combinar tamaños de kernel y preservar el procesamiento por canal antes de la fusión mejora la discriminación de gestos [32].

Por otro lado, Rezaee et al. plantean un pipeline de DL que convierte ventanas de sEMG en espectrogramas (STFT), extrae rasgos espaciales con un codificador MobileNetV2 dentro de una U-Net y modela dependencias temporales con un BiLSTM. Reportan validación cruzada (5-fold) en múltiples bases: accuracy de 92.40 % (BioPatRec DB1), 91.48 % (BioPatRec DB3) y 90.54 % (Mendeley). Con un promedio global cercano a 90.7 % y robustez frente a ruido. El estudio incluye

análisis de sensibilidad a tamaño y solapamiento de ventana [33].

Por último, la revisión de Xiong et al. sintetiza arquitecturas típicas (CNN 1D/2D/3D, RNN/LSTM/GRU, autoencoders), esquemas de preprocesamiento y tareas como clasificación de movimientos, además de retos como generalización intersesión e intersujeto. Adicionalmente, destaca la ventaja del aprendizaje end to end para extraer representaciones jerárquicas [34].

Cuadro 5.1: Resumen de trabajos relacionados en sEMG para reconocimiento de gestos

Artículo	**Modelos**	**Características extraídas**	**Preprocesamiento**	**Métricas**
Ninapro (Scientific Data, 2014) [31]	k-NN, SVM, Random Forest, LDA	RMS, TD (estadísticas en tiempo), HIST, mDWT; combinación de todas	50 gestos; división por repeticiones cuidando sesgos; recomendación de resultados balanceados	*Accuracy* promedio: 75.32 % (DB1, RF+All), 75.27 % (DB2, RF+All), 46.27 % (DB3, SVM+All)
Ding et al. (Sustainability, 2018) [32]	CNN multiescala paralela (kernels grandes; fusión tardía)	Aprendizaje *end-to-end* (sin handcrafting)	Ninapro DB2; procesamiento por canal respetando independencia muscular	*Accuracy* 78.86 % (DB2); supera baselines (p. ej., 76.1 %, 77.8 %)
Rezaee et al. (Sci. Reports, 2024) [33]	U-Net + codificador MobileNetV2 + BiLSTM (opt. bayesiana)	Espectrogramas (STFT) como entrada; *deep features* espaciales/temporales	Ventaneo y solape; evaluación 5-fold repetida; análisis de ruido; ajuste por dataset; enfoque *edge*	*Accuracy*: 92.40 % (BioPatRec DB1), 91.48 % (DB3), 90.54 % (Mendeley); promedio ≈ 90.7 %
Xiong et al. (IEEE/CAA J. Autom. Sinica, 2021) [34]	Revisión: CNN, RNN/LSTM/GRU, AE, DBN, mixtos; transferencia	síntesis de literatura	Resumen de esquemas de preprocesado y tareas; retos intersujeto/intersesión y robustez	revisión y comparación

Capítulo 6

Materiales y Métodos

6.1. Metodología

En este proyecto se consideraron dos enfoques complementarios, la Metodología de Ingeniería y la metodología CRISP-DM. Cada una cumple un propósito distinto dentro del desarrollo del sistema y ambas permiten abordar de forma estructurada tanto los componentes técnicos como el análisis basado en datos.

La Metodología de Ingeniería es un enfoque estructurado para el desarrollo de sistemas técnicos. Sus etapas incluyen el análisis de requisitos, el diseño de la arquitectura, la implementación, las pruebas y el mantenimiento [35, 36]. Se caracteriza por su énfasis en la confiabilidad, la trazabilidad y la calidad técnica de los componentes.

La metodología CRISP-DM (Cross-Industry Standard Process for Data Mining) es uno de los marcos más utilizados en ciencia de datos. Surgió a finales de los años 90 dentro de un consorcio europeo conformado por Daimler-Chrysler, NCR, SPSS y OHRA, con el objetivo de establecer un proceso estándar, abierto y reproducible para proyectos basados en datos [37, 38].

CRISP-DM organiza el proceso en seis fases interdependientes [39], que se explican a continuación: (i) Comprensión del negocio, donde se definió el objetivo de clasificar movimientos asociados al riesgo biomecánico del síndrome del túnel carpiano y los criterios de evaluación; (ii) Comprensión de los datos, que incluyó el análisis exploratorio de la base NinaPro DB2, la revisión de la distribución de clases y la identificación de los movimientos de interés; (iii) Preparación de los datos, que abarcó el filtrado de las señales (20–450 Hz y filtro notch de 50 Hz), la normalización Z-score, la segmentación en ventanas y la extracción de 144 características para modelos clásicos; (iv) Modelado, fase en la que se entrenaron algoritmos de Machine Learning y Deep Learning, incorporando técnicas de balanceo como SMOTE, ADASYN, data augmentation y Focal Loss; (v) Evaluación, donde se compararon las métricas de desempeño (AUC, F1-score, sensibilidad) para seleccionar el modelo más estable; (vi) Despliegue, en el que se integraron los modelos exportados dentro del backend en Flask y la aplicación web desarrollada para realizar inferencia sobre señales nuevas.

A diferencia de metodologías rígidas, CRISP-DM permite iterar entre fases cuando se identifican problemas en la calidad de los datos, en el balanceo o en el desempeño de los modelos. Esta flexibilidad fue clave para ajustar el pipeline durante los experimentos y mejorar la estabilidad de

los modelos.

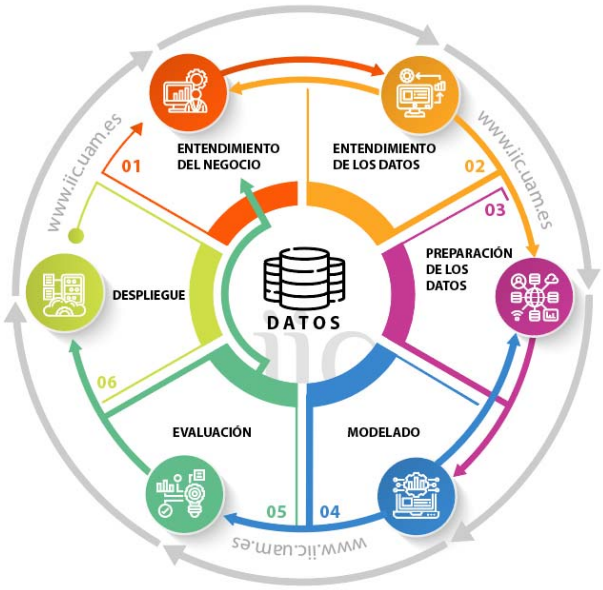


Figura 6.1: Metodología CRISP-DM

Cuadro 6.1: Evaluación comparativa de metodologías para el proyecto

Criterio	**Peso (%)**	**Design Thinking**	**CRISP-DM**	**Metodología de Ingeniería**
Orientación al usuario final	15	9	7	5
Enfoque en Machine Learning	20	3	9	6
Documentación	15	5	9	8
Reproducibilidad	15	4	9	8
Escalabilidad	15	5	8	7
Facilidad de uso	20	8	7	6
Puntaje total ponderado (%)	100	**56.5 %**	**81.5 %**	**66.0 %**

A partir de esta comparación, CRISP-DM obtuvo el puntaje más alto (81.5 %), evidenciando fortalezas clave como la reproducibilidad, la calidad de la documentación y su orientación natural al procesamiento de datos y entrenamiento de modelos. Por ello, se adoptó como metodología principal del proyecto para conducir todas las fases relacionadas con análisis de datos, entrenamiento de modelos y validación.

6.2. Diseño metodológico

Se fundamenta en la aplicación del modelo CRISP-DM, adaptado al contexto biomédico del análisis de señales de electromiografía de superficie (sEMG) para la clasificación de movimientos de la mano asociados al síndrome del túnel carpiano (STC).

El modelo CRISP-DM divide el proceso en las seis fases principales que se explicaran a continuación:

6.2.1. Entendimiento del negocio

Como se explicó en secciones anteriores, existe la necesidad de desarrollar una herramienta no invasiva y de bajo costo que permita identificar tempranamente patrones de movimiento asociados al riesgo de desarrollar STC. Desde esta perspectiva, el presente proyecto busca contribuir a la prevención y diagnóstico temprano del STC mediante el desarrollo de una herramienta computacional no invasiva capaz de analizar señales de electromiografía de superficie (sEMG) y clasificar los movimientos de la mano relacionados con el riesgo biomecánico.

La población objetivo del proyecto está conformada por personas expuestas a factores de riesgo asociados al síndrome del túnel carpiano, sin requerir un diagnóstico previo. En particular, se consideran individuos sanos que realizan de manera recurrente movimientos de flexión, extensión o desviación de la muñeca, reconocidos como biomecánicamente riesgosos.

En la fase de análisis inicial se identificó la ausencia de sistemas portátiles y automatizados que integren señales de electromiografía de superficie con modelos de aprendizaje automático para la clasificación de estos movimientos. Sin embargo, la disponibilidad de bases de datos públicas, como NinaPro, proporciona una base adecuada para el entrenamiento y validación de modelos, mitigando las limitaciones logísticas de adquisición de señales. En este contexto, el proyecto propone el procesamiento de señales sEMG y el entrenamiento de modelos capaces de discriminar entre movimientos de riesgo y movimientos seguros.

6.2.2. Comprensión de los datos

En esta fase se identificó, recopiló y analizó los conjuntos de datos más adecuados para alcanzar el objetivo del proyecto. Para seleccionar la base de datos más apropiada, se realizó una revisión exhaustiva de repositorios públicos de sEMG utilizados en la literatura científica reciente. Se identificaron seis bases de datos relevantes: NinaPro, CapgMyo, CSL-HDEMG, SEEDS, HIT-SimCo, Hyser y putEMG.

Posteriormente, se aplicó el método de decisión multicriterio Proceso Analítico Jerárquico (AHP), con el objetivo de realizar una selección sistemática basada en los criterios técnicos definidos abajo:

- **Cantidad de señales:** mejora la generalización del modelo y reduce el riesgo de overfitting.
- **Movimientos de interés:** determina la aplicabilidad de la base de datos al problema de investigación.
- **Adquisición de las señales (frecuencia de muestreo):** influye directamente en el desempeño de los algoritmos.
- **Documentación y respaldo:** los protocolos claros aseguran la reproducibilidad de los experimentos.
- **Arreglo de electrodos:** el layout determina la información espacial disponible.

En la Tabla 6.2 se presenta la comparación entre las principales bases de datos evaluadas. Los criterios analizados incluyeron el número de sujetos, la inclusión de movimientos de muñeca, la cantidad de electrodos, la frecuencia de muestreo, la disponibilidad de variables cinemáticas o de fuerza, la existencia de sesiones cross day, y el tipo de paradigma experimental.

Cuadro 6.2: Comparación de bases de datos de sEMG revisadas.

Criterio	**NinaPro**	**CapgMyo**	**CSL-HDEMG**	**SEEDS**	**HIT-SimCo**	**Hyser**	**putEMG**
No. Sujetos	10–40	10–18	5	25	8	20	44
Amputados	Sí	No	No	No	No	No	No
Gestos / Mov.	53	8	27	13	11	34	8
Mov. Muñeca	Sí	No	No	No	Sí	Sí	Sí
Electrodos	16	128HD	192HD	126HD+8	8	256HD	24
Frecuencia (Hz)	100-2000	1000	2048	2048	1926	2048	5120
Kinem./Fuerza	Kinem	No	Kinem	Kinem	Fuerza	Fuerza	Fuerza + video
Cross-day	No	Sí	No	No	Sí	Sí	Sí
Paradigma	Gestos + control	Reconocer	Reconocer	Gestos + control	Gestos + control	Gestos + control	Gestos + fuerza

Como se observa, sólo algunas bases ofrecen tanto movimientos de muñeca como documentación completa y parámetros de adquisición adecuados. Durante el proceso, algunas bases de datos fueron descartadas debido a que no cumplían con los criterios mínimos de aplicabilidad al problema de investigación. Por ejemplo:

- **putEMG**: sólo contenía movimientos de flexión y extensión, sin desviaciones ni combinaciones funcionales, por lo que fue descartada.

- **HIT-SimCo**: no permitió el acceso a los archivos, y no se logró establecer comunicación con los autores.
- **CapgMyo**, **SEEDS** y **CSL-HDEMG**: se centraban exclusivamente en movimientos de dedos, sin incluir movimientos de muñeca, esenciales para el análisis del STC.

De este modo, se concentró el análisis en las bases que cumplían los requisitos de calidad y relevancia biomecánica, las cuales fueron comparadas mediante AHP para determinar la más adecuada.

Una vez identificadas las bases de datos candidatas, se procedió a explorar sus características internas y formatos. Se verificó la disponibilidad de metadatos (etiquetas de gestos, información de sujetos, protocolos experimentales) y la compatibilidad con entornos de análisis como Python y MATLAB. Las bases con estructuras abiertas, documentación técnica y diversidad de movimientos (incluyendo flexión, extensión y desviaciones) fueron priorizadas para su uso en el modelado posterior.

La calidad de las bases se evaluó mediante los criterios del AHP, los cuales fueron ponderados en función de su relevancia para el desarrollo del modelo de clasificación. En la Tabla 6.3 se resume el resultado del proceso de priorización, donde la base con mayor puntaje fue seleccionada para las siguientes fases de la metodología.

Cuadro 6.3: Resultados del proceso AHP para selección de base de datos.

Criterio	**Peso relativo (%)**	**Descripción del impacto**
Cantidad de señales	25	Mejora la generalización y reduce el riesgo de sobreajuste.
Movimientos de interés	30	Determina la aplicabilidad de la base de datos al problema de investigación.
Frecuencia de muestreo	20	Influye directamente en el desempeño de los algoritmos de aprendizaje automático.
Documentación y respaldo	15	Los protocolos claros aseguran la reproducibilidad de los experimentos.
Arreglo de electrodos	10	El *layout* determina la información espacial disponible.
Base seleccionada	**NinaPro (0,49 puntaje priorización)**	

La base de datos NinaPro obtuvo la mayor puntuación total, siendo seleccionada para las fases posteriores del proceso CRISP-DM. Su elección se justificó por su amplia cobertura de movimientos, disponibilidad de registros de muñeca, accesibilidad pública, documentación completa y diversidad de sujetos sanos. Además, su estructura modular en sub datasets permite una selección específica

de gestos de interés relacionados con los movimientos biomecánicos del síndrome del túnel carpiano.

La base de datos NinaPro (Non-Invasive Adaptive Prosthetics Database) no corresponde a un único conjunto de registros, sino a un compendio de múltiples bases de datos denominadas DB1–DB10, cada una orientada a distintos objetivos experimentales y tipos de sujetos (intactos o amputados). Cada sub base incluye diferentes configuraciones de electrodos, protocolos de movimiento y frecuencias de muestreo.

Según lo descrito en [40], las bases con mayor aplicabilidad para estudios de control de muñeca y análisis biomecánico son aquellas con sujetos sanos, alta frecuencia de muestreo (2 kHz), y configuración homogénea de electrodos. En la Tabla 6.4 se presenta un resumen comparativo de las sub-bases más relevantes.

Cuadro 6.4: Comparación de sub-bases de datos NinaPro. Adaptado de Chang et al., 2020.

Dataset	**Sanos**	**Amputados**	**Canales EMG**	**Config.**	**No. gestos**	**Repet.**	**Freq. (Hz)**
DB2	40	0	12	A	49	6	2000
DB3	0	11	12	A	49	6	2000
DB4	10	0	12	A	52	6	2000
DB5	10	0	16	B	52	6	200
DB6	10	0	16	B	7	12	2000
DB7	20	2	12	A	40	6	2000

De las bases listadas, las sub bases DB2, DB4 y DB7 cumplen con los criterios de calidad de señal, densidad de electrodos y disponibilidad de sujetos sanos. Sin embargo, se seleccionó **NinaPro DB2** debido a que:

- Presenta el mayor número de sujetos (40 individuos sanos).
- Posee una frecuencia de muestreo de 2000 Hz, lo que garantiza una resolución temporal adecuada para el análisis espectral de señales sEMG.
- Incluye un conjunto de movimientos con alta relevancia biomecánica para el síndrome del túnel carpiano.

Cada sujeto cuenta con un archivo en formato .mat que contiene las siguientes variables sincronizadas:

- **subject**: número identificador del participante.
- **exercise**: número de ejercicio (A, B o C).

- **emg (12 columnas)**: señales de sEMG obtenidas de 12 electrodos Delsys Trigno. Los canales 1–8 están dispuestos equidistantemente alrededor del antebrazo a la altura de la articulación radiohumeral, los canales 9–10 capturan la actividad de los músculos flexor digitorum superficialis y extensor digitorum superficialis y los canales 11–12 corresponden a los músculos biceps brachi y triceps brachi.
- **acc (36 columnas)**: datos de acelerometría triaxial de los 12 electrodos.
- **glove (22 columnas)**: señal no calibrada del guante cibernético (CyberGlove II), proporcional a los ángulos articulares de los dedos y la muñeca.
- **stimulus** y **restimulus**: stimulus es la etiqueta de movimiento durante la ejecución y las fases de reposo, la cual fue corregida manualmente y anexada como la variable restimulus.
- **repetition**: repetition contiene el número de repetición del gesto (de 1 a 6), también fue manualmente corregida y anexada como la variable rerepetition.

Cada movimiento tiene una duración de 5 segundos, seguida de 3 segundos de descanso, y se repite seis veces por sujeto. La adquisición se realizó con los participantes ejecutando los movimientos de la mano derecha, a partir un video guía en una pantalla, como se observa en la figura 6.2.

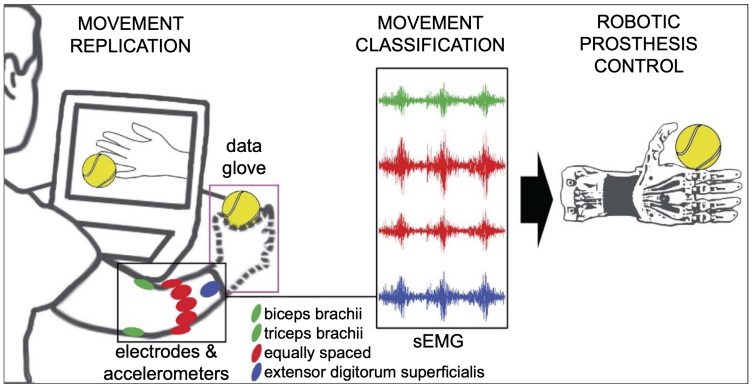


Figura 6.2: Protocolo de adquisición Ninapro

Dentro de la DB, los movimientos se agrupan en tres conjuntos o ejercicios, como se ve en la figura 6.3 y se explica a continuación:

- **A** - Movimientos básicos de los dedos.
- **B** - Movimientos funcionales de la muñeca y prensión.
- **C** - Patrones de fuerza.

Para este trabajo se seleccionó exclusivamente el Ejercicio B, ya que contiene los movimientos de interés relacionados con el síndrome del túnel carpiano, en particular los gestos de:

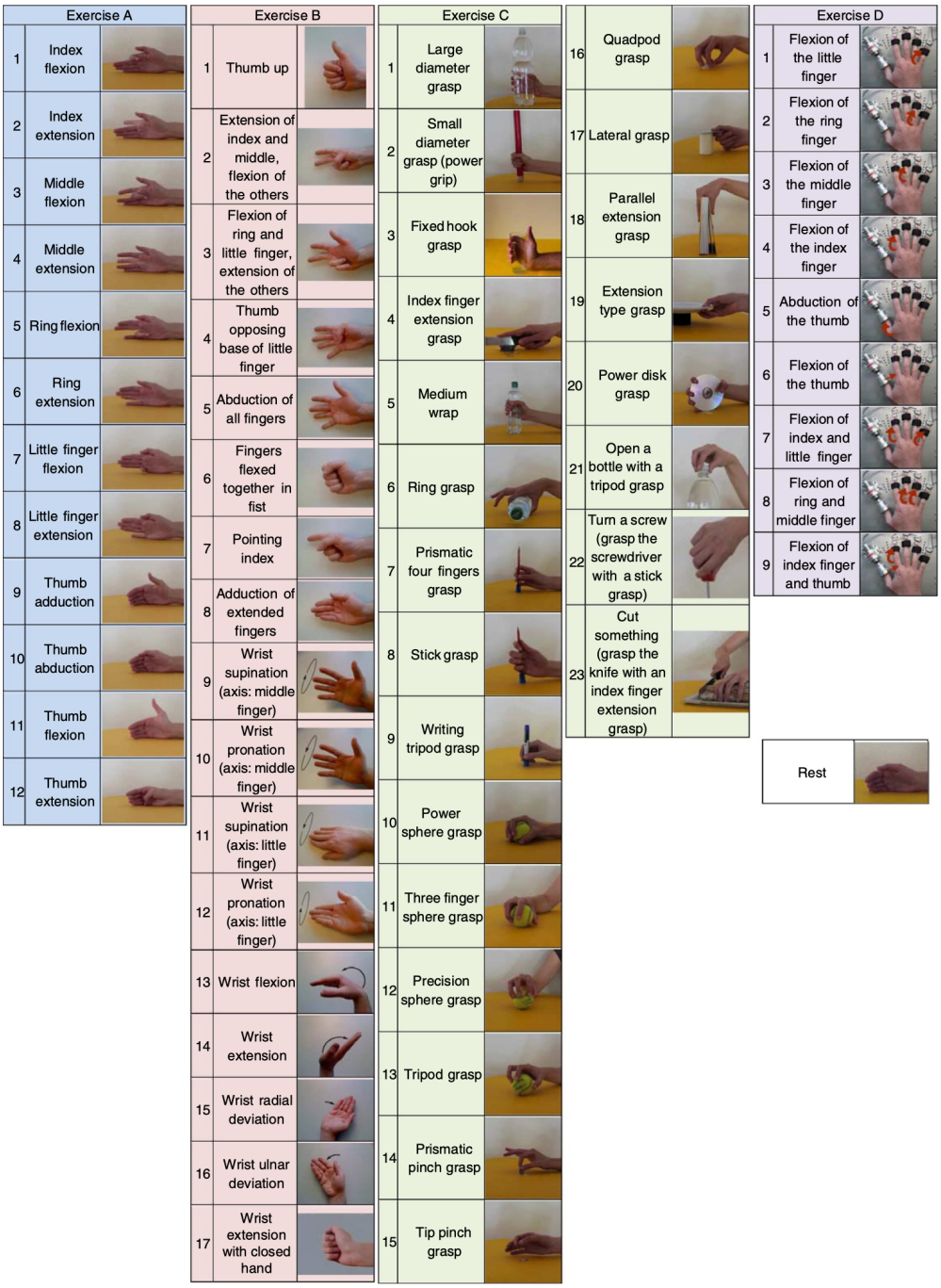


Figura 6.3: Movimientos por conjunto de ejercicios

- **Movimiento 13:** Flexión de muñeca.

- **Movimiento 14:** Extensión de muñeca.
- **Movimiento 15:** Desviación radial.
- **Movimiento 16:** Desviación cubital.

Estos cuatro gestos fueron seleccionados por su relevancia clínica y biomecánica, dado que representan los principales patrones de movilidad asociados a la tensión del nervio mediano y a la sintomatología del túnel carpiano, de acuerdo a lo revisado en la literatura y consulta con profesionales.

6.2.3. Preparación de los datos

Tras la selección de la base NinaPro DB2, los archivos originales fueron descargados desde el repositorio oficial en formato .mat. Para estructurar el flujo de trabajo, se desarrolló un script en Python denominado split_dataset_db2.py, cuyo propósito fue separar los archivos en dos subconjuntos:

- **Train:** Contiene los archivos originales completos de cada sujeto, destinados al entrenamiento, validación y testeo de los modelos en el cuaderno de Colab.
- **Test:** Contiene archivos individuales correspondientes a un único movimiento y repetición por sujeto, utilizados para la fase de despliegue en la aplicación final.

El script automatiza este proceso creando directorios dedicados (train/ y test/), copiando los archivos completos a la carpeta de entrenamiento y extrayendo repeticiones específicas (por defecto, la repetición 5) para conformar las muestras de prueba. Cada archivo de prueba incluye las variables: emg, subject, exercise, frequency, rerepetition y movement, excluyendo las etiquetas temporales restimulus para evitar sesgos en la evaluación.

Luego se realizó un análisis espectral de las señales sin filtrar para caracterizar el comportamiento de los movimientos y fundamentar las decisiones de preprocesamiento. Este análisis incluyó el cálculo de la densidad espectral de potencia (PSD) mediante transformada de Fourier para identificar los rangos de frecuencia dominantes en las señales, la extracción de características espectrales por canal (frecuencia pico, frecuencia media y potencia total). Esta información es con el fin de asegurar que el pipeline de procesamiento preserve la información discriminativa identificada en este análisis exploratorio.

El pipeline de procesamiento fue implementado en Google Colab mediante el archivo heclamostc.ipynb, donde se definieron las funciones de filtrado y normalización aplicadas a las señales electromiográficas. Las señales se cargaron desde la carpeta train/ y fueron sometidas a un doble filtrado:

1. **Filtro pasa banda Butterworth:** para eliminar componentes de baja frecuencia (movimientos lentos o artefactos de base) y altas frecuencias no fisiológicas.
2. **Filtro Notch:** para atenuar el ruido eléctrico producido por la red de alimentación.

Posteriormente, las señales fueron normalizadas mediante un escalado Z-score, lo cual garantiza que cada canal de EMG tenga media cero y desviación estándar unitaria. Esta normalización es esencial para estabilizar el entrenamiento de redes neuronales y evitar que los modelos ML se vean dominados por canales de alta amplitud.

Una vez filtradas y normalizadas, las señales fueron segmentadas en ventanas deslizantes. Cada ventana fue etiquetada según la moda del estímulo dentro del intervalo, conservando unicamente las ventanas con el 90 % de la misma etiqueta. Se excluyeron las ventanas correspondientes a la clase rest (sin movimiento) para centrarse únicamente en las acciones voluntarias.

Los movimientos se codificaron en dos categorías:

- **Clase 0 (Safe):** movimientos sin riesgo ergonómico.
- **Clase 1 (Risk):** movimientos asociados al síndrome del túnel carpiano, correspondientes a los gestos 13–16 (flexión, extensión, desviación radial y desviación cubital).

A partir de estas ventanas, se construyeron dos estructuras de datos paralelas:

- **Features (ML):** cada ventana fue convertida en un vector de características calculadas en los dominios temporal, frecuencial y wavelet. Se extrajeron métricas como Mean Absolute Value (MAV), Waveform Length (WL), Zero Crossing (ZC), Slope Sign Changes (SSC), Root Mean Square (RMS), frecuencias media y mediana, y energías por nivel wavelet.
- **Secuencias (DL):** las ventanas se mantuvieron en formato temporal tridimensional samples $\times$ time $\times$ channels para ser procesadas directamente por arquitecturas profundas.

Durante la segmentación se evidenció un desbalance entre las clases “Safe” y “Risk”. Para mitigar este problema y evitar sesgos durante el entrenamiento, se implementaron las siguientes estrategias de balanceo:

1. **SMOTE (Synthetic Minority Oversampling Technique):** genera muestras sintéticas de la clase minoritaria interpolando observaciones reales cercanas en el espacio de características. Esta técnica se empleó sobre los vectores de características del conjunto de entrenamiento para modelos de ML.
2. **ADASYN (Adaptive Synthetic Sampling):** variante de SMOTE que genera más muestras sintéticas en las regiones donde la clase minoritaria es más difícil de aprender. En lugar de producir un sobremuestreo uniforme, ADASYN detecta automáticamente las zonas donde la densidad de ejemplos minoritarios es baja y sintetiza datos adicionales allí, promoviendo un aprendizaje más equilibrado y robusto. Esta técnica se empleó sobre los vectores de características del conjunto de entrenamiento para modelos de ML.

3. **Focal Loss:** función de pérdida diseñada para abordar desbalance severo en problemas de clasificación binaria, reduciendo el impacto de las muestras fácilmente clasificables. Introduce dos parámetros: γ (factor de enfoque), que amplifica la penalización hacia ejemplos difíciles, y α, que controla el peso relativo de la clase minoritaria. Fue utilizada en los modelos de Deep Learning para contrarrestar la predominancia de la clase segura durante el entrenamiento.

4. **Data Augmentation para DL:** conjunto de transformaciones aplicadas directamente sobre las ventanas de la señal EMG para aumentar la diversidad del conjunto de entrenamiento sin alterar la distribución real del problema. Entre las operaciones utilizadas se incluyen *jittering* (ruido aditivo gaussiano), *scaling* (amplificación o atenuación global de la señal) y *time warping* (deformaciones temporales suaves mediante interpolación).

Las estrategias fueron evaluadas según el comportamiento del conjunto de entrenamiento y validación, seleccionando la que ofreciera mejor estabilidad y desempeño sin sobreajuste.

Finalmente, los datos fueron empaquetados en estructuras optimizadas para su posterior uso en modelado:

- `train/`, `val/` y `test/` para particiones independientes de acuerdo a la repetición. Es decir, se usaron las repeticiones 1,3,4,6 para train (al rededor del 70 % de los datos), 2 para validation (15 %) y 5 para test (15 %), de acuerdo a lo recomendado en la guía de uso de la base de datos [31].
- `features_data` para los modelos ML.
- `windowed_data` para los modelos DL.

6.2.4. Elección de los modelos

A partir de la revisión de la literatura, se identificaron los modelos de clasificación más empleados en el reconocimiento de gestos mediante sEMG tanto en enfoques tradicionales de ML como en DL, ver tabla de trabajos relacionados 5.1.

El prototipado inicial se realizó en MATLAB mediante el Classification Learner, lo que permitió comparar múltiples algoritmos estándar (SVM, KNN, árboles, redes neuronales superficiales). A partir de esos resultados preliminares, se seleccionaron los dos modelos clásicos con mejor desempeño: Ensemble Subspace KNN y Random Forest.

En el caso de los modelos clásicos, se priorizaron aquellos con bajo costo computacional, mientras que para los modelos de DL se consideraron arquitecturas ampliamente utilizadas en el procesamiento de señales fisiológicas. Los modelos finalmente seleccionados fueron:

- **Ensemble Subspace KNN:** consiste en múltiples clasificadores KNN entrenados sobre subespacios aleatorios de las características (random subspace method). Cada estimador em-

plea un KNN con métrica euclidiana y pesos inversamente proporcionales a la distancia. Ver estructura en la figura 6.4.

- **Random Forest:** clasificador basado en un conjunto de árboles de decisión entrenados con muestreo aleatorio de datos y subconjuntos de características. Su estructura conceptual se muestra en la figura 6.5.
- **CNN + LSTM con Attention:** arquitectura híbrida que combina bloques convolucionales 1D para la extracción jerárquica de patrones locales de activación muscular, seguidos de capas LSTM que modelan dependencias temporales. Sobre la salida recurrente se aplica un mecanismo de attention que pondera dinámicamente los instantes más informativos de la ventana. Adicionalmente, se emplean regularización L2 y dropout para mitigar el sobreajuste. La estructura general sin capas attention se presenta en la figura 6.6.
- **Bidirectional LSTM (BiLSTM) con Attention:** variante que procesa la secuencia en ambas direcciones (adelante y atrás), incrementando la capacidad del modelo para capturar patrones temporales complejos en señales EMG. Sobre la concatenación de las direcciones se incorpora un módulo de attention que enfatiza las regiones temporalmente relevantes. Este modelo incluye normalización por lotes, regularización L2 y dropout para mejorar la generalización. Su estructura general sin capas attention se muestra en la figura 6.7.

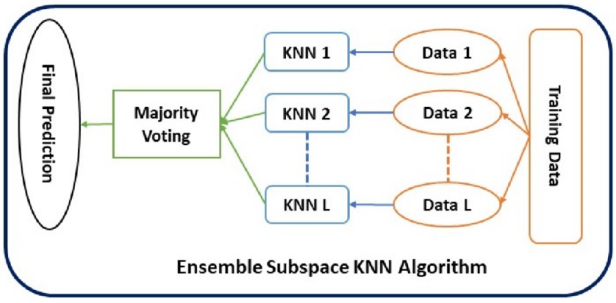


Figura 6.4: Arquitectura básica de un ensemble KNN

El diseño de pruebas se basó en la división establecida durante la fase de preparación de datos: repeticiones 1, 3, 4, 6 para entrenamiento, repetición 2 para validación y repetición 5 para prueba, garantizando la independencia de las muestras.

Los modelos ML fueron ajustados usando el conjunto de características previamente calculadas (MAV, WL, RMS, ZC, SSC, etc.), mientras que los modelos DL se entrenaron directamente sobre las secuencias de ventanas.

Cada modelo se configuró con los hiperparámetros óptimos obtenidos por random search y documentados en la Tabla 6.5.

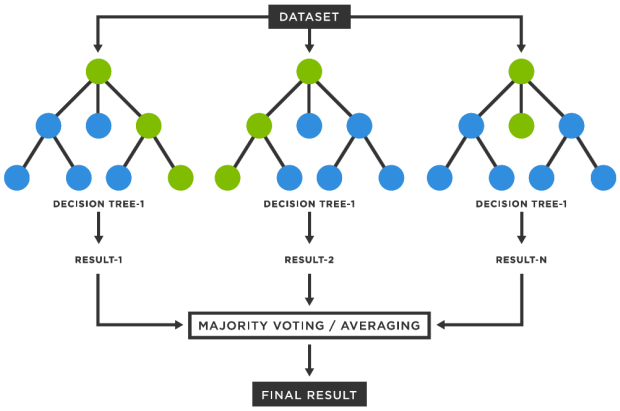


Figura 6.5: Arquitectura básica de Random Forest

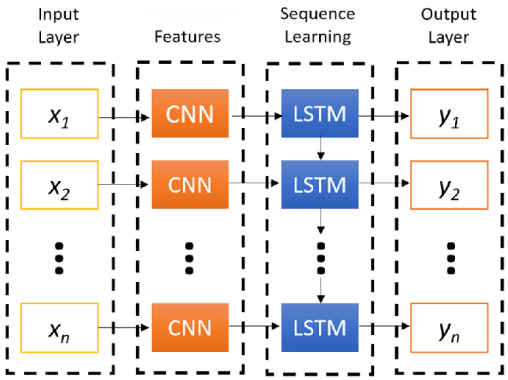


Figura 6.6: Arquitectura básica de una CNN + LSTM

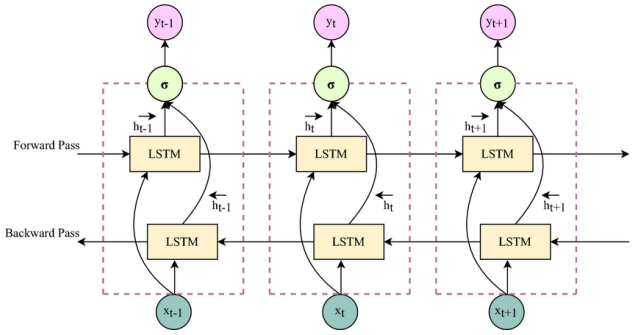


Figura 6.7: Arquitectura básica de una BiLSTM

6.2.5. Evaluación

Durante esta fase se llevó a cabo un proceso sistemático de evaluación, que comprendió cinco experimentos comparativos y un análisis detallado de métricas y visualizaciones de desempeño. Para analizar el impacto de las estrategias de balanceo aplicadas durante el entrenamiento, se realizaron tres configuraciones experimentales independientes:

1. **Experimento 1 — Sin balanceo:** los modelos fueron entrenados únicamente con las ventanas resultantes de la segmentación, sin aplicar técnicas de sobremuestreo ni ajustes de pérdida. Este escenario establece la línea base del desempeño de los modelos ante el desbalance natural entre las clases "Safe" y "Risk".

2. **Experimento 2 — SMOTE para ML + Data Augmentation y Focal Loss para DL:** en los modelos de Machine Learning se aplicó SMOTE para sintetizar muestras adicionales de la clase minoritaria en el espacio de características. Para los modelos de Deep Learning se empleó data augmentation basado en jittering, scaling y time warping, junto con la función de pérdida Focal Loss.

3. **Experimento 3 — ADASYN para ML + Data Augmentation y Focal Loss para DL:** los modelos de ML utilizaron ADASYN, que genera más muestras sintéticas en regiones de baja densidad de la clase minoritaria. Paralelamente, los modelos DL se entrenaron combinando data augmentation con Focal Loss.

4. **Experimento 4 — ADASYN para ML + Data Augmentation para DL:** en este caso, ADASYN se aplicó únicamente a los modelos de ML, mientras que los modelos DL utilizaron exclusivamente data augmentation, sin Focal Loss.

5. **Experimento 5 — ADASYN para ML + Focal Loss para DL:** aquí, los modelos de ML fueron balanceados mediante ADASYN, mientras que los modelos DL se entrenaron con Focal Loss, sin data augmentation.

Cada experimento fue evaluado con los mismos conjuntos de validación y prueba, asegurando la consistencia en las comparaciones y evitando la contaminación cruzada de datos.

El rendimiento de los modelos se cuantificó mediante las métricas clásicas de clasificación binaria: accuracy, precision, recall y F1-score. Cada una se calculó a partir de la matriz de confusión, que resume las predicciones correctas e incorrectas según las categorías reales.

La matriz de confusión se define como:

$$\begin{bmatrix} TP & FN \\ FP & TN \end{bmatrix}$$

donde:

- TP (True Positives): movimientos de riesgo correctamente identificados.
- FN (False Negatives): movimientos de riesgo clasificados erróneamente como seguros.
- FP (False Positives): movimientos seguros clasificados como de riesgo.
- TN (True Negatives): movimientos seguros correctamente identificados.

A partir de estos valores se derivan las siguientes métricas:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$\text{F1-Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

El Accuracy evalúa el porcentaje total de predicciones correctas, Precision mide la confiabilidad del modelo al identificar movimientos de riesgo, Recall refleja la capacidad del modelo para detectar todos los casos de riesgo presentes, y el F1-score pondera simultáneamente precisión y sensibilidad, resultando útil cuando las clases están desbalanceadas.

Además, se utilizó el área bajo la curva ROC (AUC) como métrica adicional para evaluar la capacidad discriminativa global del modelo, considerando todos los posibles umbrales de decisión. El AUC se define como el área bajo la curva que grafica la tasa de verdaderos positivos (TPR) frente a la tasa de falsos positivos (FPR).

6.2.6. Despliegue

La implementación final del proyecto consiste en una aplicación web interactiva orientada a la clasificación de movimientos de la mano asociados al síndrome del túnel carpiano, permitiendo la carga de señales sEMG en formato .mat y la obtención de predicciones de riesgo mediante modelos de aprendizaje automático previamente entrenados. La arquitectura adoptada sigue un esquema cliente-servidor, en el cual el frontend gestiona la interacción con el usuario y el backend centraliza el preprocesamiento de señales, la carga de modelos y la inferencia.

6.2.6.1. Arquitectura general del sistema

El sistema se estructura en tres componentes principales, como se observa en 6.8, los cuales son: (i) un backend implementado en Flask, responsable de la lógica de procesamiento e inferencia; (ii) un frontend desarrollado con HTML, CSS y JavaScript, encargado de la visualización y control de la aplicación; y (iii) un repositorio de modelos entrenados que incluye clasificadores clásicos y profundos junto con los objetos auxiliares necesarios para el preprocesamiento. Esta separación permite desacoplar la interfaz del motor de inferencia y facilita la extensibilidad del sistema.

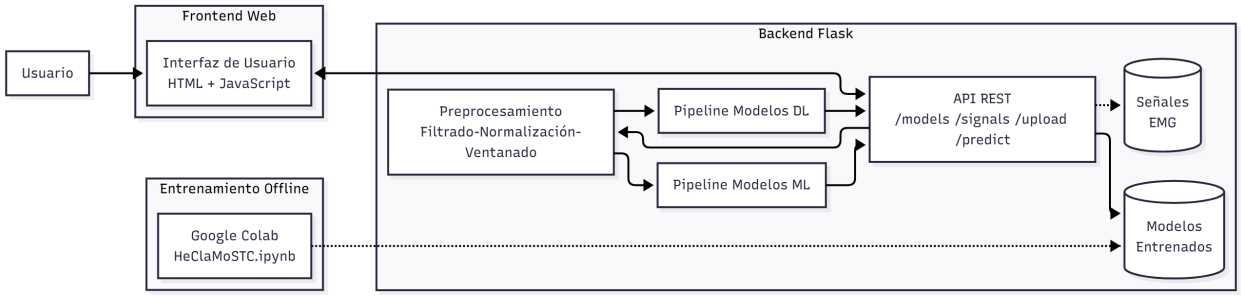


Figura 6.8: Diagrama general del sistema

6.2.6.2. Backend: configuración y componentes

El backend fue implementado en Python utilizando el framework Flask, complementado con mecanismos de control de acceso para habilitar la comunicación con el frontend. Para el procesamiento numérico y de señales se emplearon las librerías NumPy y SciPy, mientras que la lectura de archivos MAT se realizó mediante funciones de carga específicas. Los modelos de aprendizaje automático se cargan desde archivos serializados en formato PKL, mientras que los modelos de aprendizaje profundo se integran a partir de archivos en formatos KERAS.

El backend define una clase de configuración centralizada que replica exactamente los parámetros utilizados durante el entrenamiento en Google Colab. El pipeline de preprocesamiento implementado incluye el filtrado digital de la señal sEMG, la normalización mediante el mismo escalador utilizado durante el entrenamiento, la segmentación en ventanas temporales y, según el tipo de modelo seleccionado, la extracción de características en los dominios temporal, frecuencial y tiempo-frecuencia para los modelos de aprendizaje clásico, o la preservación de las secuencias temporales para los modelos de aprendizaje profundo. Posteriormente, se realiza la predicción por ventana y se agregan los resultados para obtener una decisión final por señal.

6.2.6.3. Endpoints y lógica de inferencia

La aplicación expone distintos endpoints de tipo REST para la gestión del sistema. Un endpoint permite consultar dinámicamente los modelos disponibles, diferenciando entre modelos de aprendizaje clásico y profundo. Otro endpoint lista las señales disponibles tanto desde un directorio externo como desde señales cargadas por el usuario. Adicionalmente, se implementó un endpoint para la carga de nuevos archivos MAT, validando su formato antes de almacenarlos temporalmente en el servidor.

El endpoint principal de predicción implementa la lógica de clasificación y opera en dos modos: un modo independiente, en el que se utiliza un único modelo seleccionado por el usuario, y un modo dual, en el que se combinan dos modelos clásicos especializados en cascada. En este último caso, el sistema emplea un clasificador con alta precisión para la clase segura y un clasificador con alta sensibilidad para la clase de riesgo, tomando la decisión final en función de una lógica jerárquica. La respuesta del sistema incluye la clase final (seguro o riesgo), probabilidades promedio, número de ventanas analizadas, porcentaje de ventanas clasificadas como riesgo y metadatos asociados a la señal, además de una versión submuestreada de la señal para su visualización.

6.2.6.4. Frontend: funcionalidades e interacción

El frontend fue desarrollado utilizando HTML, CSS y JavaScript, priorizando simplicidad y compatibilidad. La interfaz permite seleccionar el modo de clasificación (modelo independiente o sistema dual), elegir el tipo y nombre del modelo, cargar archivos MAT y seleccionar señales disponibles para su análisis. La comunicación con el backend se realiza mediante peticiones asíncronas a los endpoints REST, utilizando intercambio de datos en formato JSON.

Los resultados de la clasificación se presentan mediante tarjetas visuales diferenciadas por color para movimientos seguros y de riesgo, incluyendo indicadores de confianza, probabilidades promedio y metadatos relevantes. Adicionalmente, se visualiza un gráfico de la señal sEMG submuestreada correspondiente al primer canal, lo que facilita la inspección cualitativa de la señal analizada y mejora la interpretabilidad del sistema.

Con el fin de garantizar la reproducibilidad y estabilidad del sistema, el código del backend, el frontend y los modelos entrenados se mantienen bajo control de versiones. La estructura modular del proyecto permite actualizar modelos, ajustar parámetros de preprocesamiento o extender las funcionalidades de la interfaz sin modificar el flujo principal de inferencia, sentando las bases para futuros despliegues en entornos reales o integraciones con sistemas de adquisición de señales sEMG.

6.3. Componentes

El desarrollo del presente trabajo se realizó completamente en entorno digital, ya que se enfoca en el procesamiento y análisis de señales biomédicas existentes. Por tanto, los componentes empleados corresponden a equipos y recursos computacionales que permitieron la implementación, documentación y validación del proyecto.

6.3.1. Hardware

- **Computador portátil:** Acer Nitro 5 AN515-55 con procesador Intel Core i5 de 10ª generación, una tarjeta gráfica NVIDIA (GTX 1650) y 16 GB de RAM.

6.3.2. Software

- **Repositorio de datos NinaPro:** La Non Invasive Adaptive Prosthetics (Ninapro) es una base de datos de acceso público que contiene un conjunto completo y multimodal de registros electromiográficos y de movimiento, recopilados con el objetivo de fomentar la investigación y el desarrollo de algoritmos de aprendizaje automático.
- **Google Colab:** Entorno de desarrollo basado en la nube que permitió la programación en Python para el procesamiento de señales, la implementación de modelos de aprendizaje automático y la visualización de resultados.
- **Overleaf:** Plataforma empleada para la redacción, organización y compilación de la tesis en formato LaTeX.
- **Microsoft Excel:** Utilizado para la realización de tablas, cálculo de ponderaciones del método *Analytic Hierarchy Process (AHP)*, y consolidación de resultados intermedios durante la fase de evaluación de modelos.
- **Google Drive:** Utilizados como plataformas de almacenamiento y respaldo de los archivos del proyecto.

6.4. Tipo de estudio

El presente trabajo corresponde a un estudio de carácter cuantitativo, orientado al desarrollo y validación de un modelo computacional para la clasificación automática de movimientos de la muñeca mediante señales de electromiografía de superficie (sEMG).

El enfoque cuantitativo se justifica en el hecho de que todas las variables utilizadas son de naturaleza numérica y medible, derivadas directamente de las señales sEMG adquiridas en la base de datos. No se emplearon variables cualitativas, ya que el objetivo principal del estudio se centra en la relación entre los patrones eléctricos musculares y los movimientos de interés biomecánico.

Cuadro 6.5: Modelos empleados y principales parámetros de configuración.

Modelo	Descripción y configuración principal
Ensemble Subspace KNN	Ensamble de 30 clasificadores KNN base con $k = 7$, métrica euclidiana y pesos por distancia. Cada estimador utiliza el 70 % de las muestras y el 80 % de las características (*max_samples* = 0.7, *max_features* = 0.8), siguiendo el esquema *random subspace*
Random Forest	Bosque aleatorio compuesto por 200 árboles de decisión. Configuración: profundidad máxima = 30, *min_samples_split* = 5, *min_samples_leaf* = 2 y selección aleatoria de características con *max_features* = `sqrt`.
CNN + LSTM con Attention	Arquitectura híbrida reducida. Incluye dos bloques convolucionales 1D: (1) Conv1D con 64 y 128 filtros (kernel = 3), seguidas de *BatchNorm*, *MaxPooling* y *Dropout*; (2) Conv1D con 256 y 256 filtros (kernel = 3), nuevamente con *BatchNorm*, *MaxPooling* y *Dropout*. A continuación, dos capas LSTM con 64 y 32 unidades, ambas con regularización L2 y *Dropout* controlado por el hiperparámetro global. Sobre la salida recurrente se aplica un mecanismo de *attention* que pondera los instantes temporales relevantes, seguido de una capa densa de 32 neuronas (ReLU, L2, *Dropout* = 0.4) y una salida sigmoide. El modelo se entrena con Adam (`learning_rate` = 0.0005), hasta 100 épocas con *Early Stopping* y *ReduceLROnPlateau*, usando *Focal Loss* (gamma = 2.5, alpha = 0.75) o *binary crossentropy* según la configuración.
Bidirectional LSTM con Attention	Modelo secuencial basado en dos capas BiLSTM reducidas con 64 y 32 unidades, respectivamente, ambas con regularización L2. Cada bloque recurrente está seguido de *BatchNorm* y, en la primera capa, de *Dropout*. La salida bidireccional alimenta un módulo de *attention* que genera pesos temporales (proyectados y repetidos sobre 64 dimensiones) para resaltar las regiones informativas de la ventana. Posteriormente se incluyen capas densas de 32 y 16 neuronas (ReLU, regularización L2 y *Dropout* = 0.4) y una salida sigmoide. Al igual que la CNN+LSTM, se entrena con Adam (`learning_rate` = 0.0005), hasta 100 épocas con *Early Stopping* y *ReduceLROnPlateau*, empleando *Focal Loss* o *binary crossentropy* según la técnica de balanceo seleccionada.

Capítulo 7

Resultados y Discusión

7.1. Descripción general de los resultados

Como se explicó anteriormente, se implementaron una serie de experimentos controlados que abarcaron tanto modelos clásicos de ML como modelos de DL. Cada experimento se basó en el conjunto de datos obtenido de la base NinaPro DB2, específicamente del Ejercicio B, correspondiente a los movimientos de flexión, extensión, desviación radial y desviación ulnar de la muñeca.

El diseño de los experimentos incluyó diferentes configuraciones de balanceo de clases: sin balanceo, SMOTE para modelos de ML, ADASYN para ML, data augmentation en DL y Focal Loss en DL. En cada una de estas configuraciones se evaluaron dos modelos clásicos (Ensemble Subspace KNN y Random Forest) entrenados sobre las 144 características extraídas manualmente, así como dos modelos de aprendizaje profundo (CNN+LSTM y BiLSTM con attention) que operan directamente sobre las ventanas de señal EMG preprocesada.

El rendimiento de los modelos fue medido a través de métricas estándar de clasificación binaria, tales como accuracy, precision, recall y F1-score, complementadas con curvas ROC y matrices de confusión para visualizar el comportamiento de las predicciones.

Posteriormente, los modelos entrenados fueron exportados e implementados en el flujo de la aplicación web desarrollada, integrándose con los módulos de preprocesamiento y visualización descritos anteriormente. Esta implementación permite que los usuarios carguen nuevas señales o utilicen las señales de prueba generadas durante la fase de preparación de datos, obteniendo la predicción del modelo correspondiente.

A partir de lo anterior se presentara el análisis de las señales y los diagramas de flujo del sistema.

7.1.1. Análisis espectral de las señales

Inicialmente se realizó una inspección visual de los espectros de frecuencia para todos los movimientos y canales, con el objetivo de identificar patrones generales en la distribución de energía. En la Figura 7.1 se presenta un ejemplo correspondiente al Sujeto 1, canal 1, repetición 1, donde se observa el comportamiento espectral completo de cada uno de los 17 movimientos. Esta visualización inicial permitió confirmar la concentración dominante de energía en el rango aproximado de 20–450 Hz, así como una mayor amplitud en los movimientos clasificados como RISK.

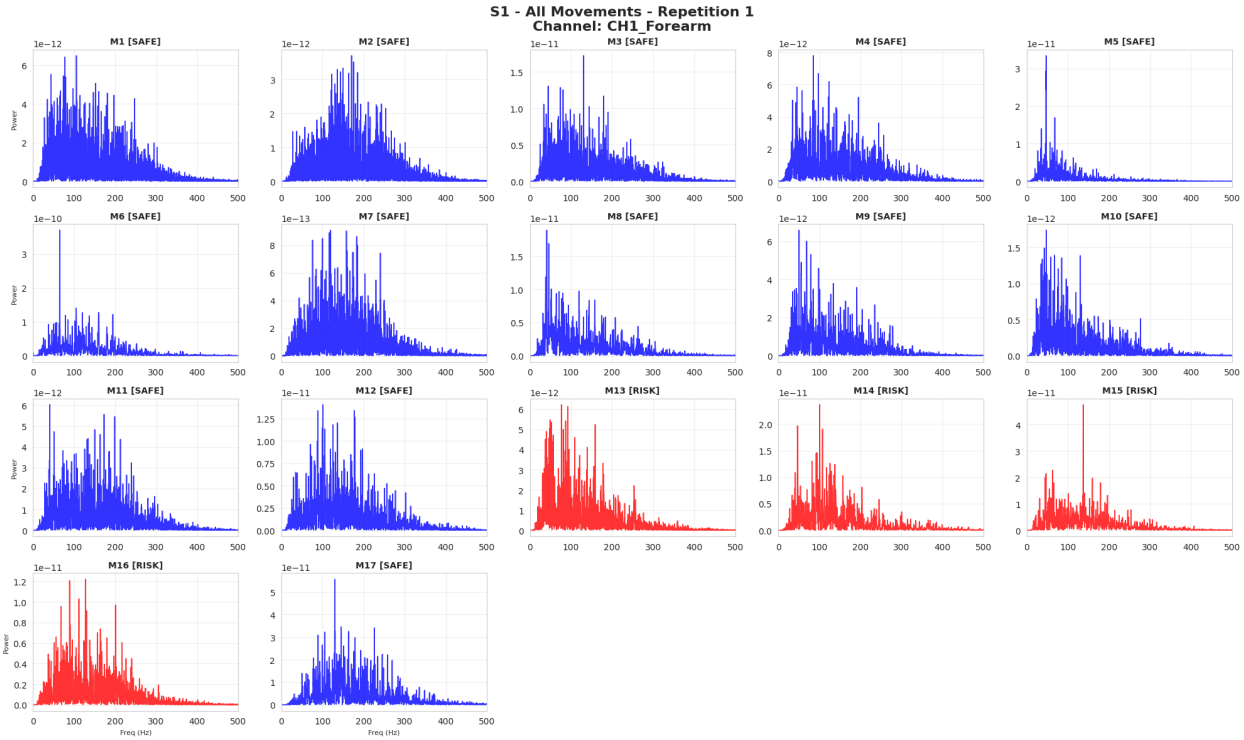


Figura 7.1: Espectro de los 17 movimientos para Sujeto 1, Canal 1 y Repetición 1

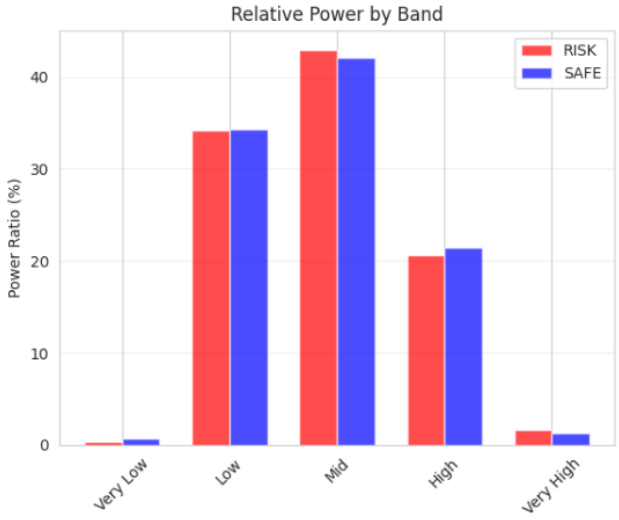


Figura 7.2: Potencia relativa por bandas de frecuencia donde Very Low: (0, 20) Hz, Low: (20, 100) Hz, Mid: (100, 200) Hz, High: (200, 400) Hz, Very High: (400, 500) Hz

A partir de estas observaciones preliminares se calcularon métricas cuantitativas de potencia por banda (Figura 7.2). El análisis confirma que el rango 20–400 Hz captura la mayor parte de la energía útil, lo que respalda la selección del rango de filtrado utilizado (band-pass 20–450 Hz y notch 50 Hz) para aprovechar las frecuencias significativas y limpiar el contenido espectral antes de la extracción de características.

Con el fin de caracterizar el comportamiento general de las señales sEMG a lo largo de todos los sujetos y movimientos, se realizó un análisis espectral global que permitió cuantificar diferencias entre las categorías SAFE y RISK en términos de frecuencia dominante, energía y estabilidad espectral.

Cuadro 7.1: Resumen global de características espectrales por categoría

Categoría	**Freq. Pico (Hz)**	**Freq. Media (Hz)**	**Potencia Total**	**Consistencia**
SAFE	$69{,}61 \pm 44{,}19$	$123{,}87 \pm 33{,}75$	$1{,}59 \times 10^{-1} \pm 1{,}23$	$0{,}4473 \pm 0{,}0099$
RISK	$74{,}26 \pm 46{,}88$	$127{,}87 \pm 35{,}37$	$1{,}71 \times 10^{-1} \pm 1{,}22$	$0{,}4481 \pm 0{,}0120$

La Tabla 7.1 resume las estadísticas globales obtenidas para los 17 movimientos. Se incluyen las métricas fundamentales: frecuencia pico, frecuencia media, potencia total y consistencia espectral promedio (correlación entre repeticiones). La base de datos incluye 13 movimientos clasificados como SAFE y 4 como RISK.

En términos generales, los movimientos RISK muestran valores ligeramente superiores en frecuencia pico (+4.65 Hz), frecuencia media (+4.0 Hz) y potencia total (+0.0119), aunque las variaciones estándar son amplias debido a la dispersión natural del sEMG. La consistencia espectral promedio entre repeticiones es prácticamente equivalente en ambas categorías ($r \approx 0{,}448$), lo que indica que, a nivel global, la estabilidad relativa del patrón espectral no constituye un factor diferenciador entre movimientos seguros y de riesgo, ver Figura 7.3.

Para evaluar si algunas tareas motoras particulares presentan diferencias notables, se analizó el comportamiento individual por movimiento. Los resultados de las Figura 7.4 y 7.5 muestran que la frecuencia media tiende a ser mayor en los movimientos RISK, particularmente en flexión, extensión y desviaciones de muñeca. Por ejemplo: Wrist flexion (M13): $132{,}82 \pm 37{,}98$ Hz y Wrist ulnar deviation (M16): $128{,}18 \pm 36{,}59$ Hz. En contraste, movimientos como Pointing index o Thumb up muestran frecuencias medias entre 116 y 122 Hz.

El análisis espectral global muestra que los movimientos de riesgo tienden a involucrar activación muscular ligeramente más intensa y de mayor frecuencia, pero las diferencias globales no son suficientemente fuertes como para separar categorías sin el apoyo de modelos supervisados.

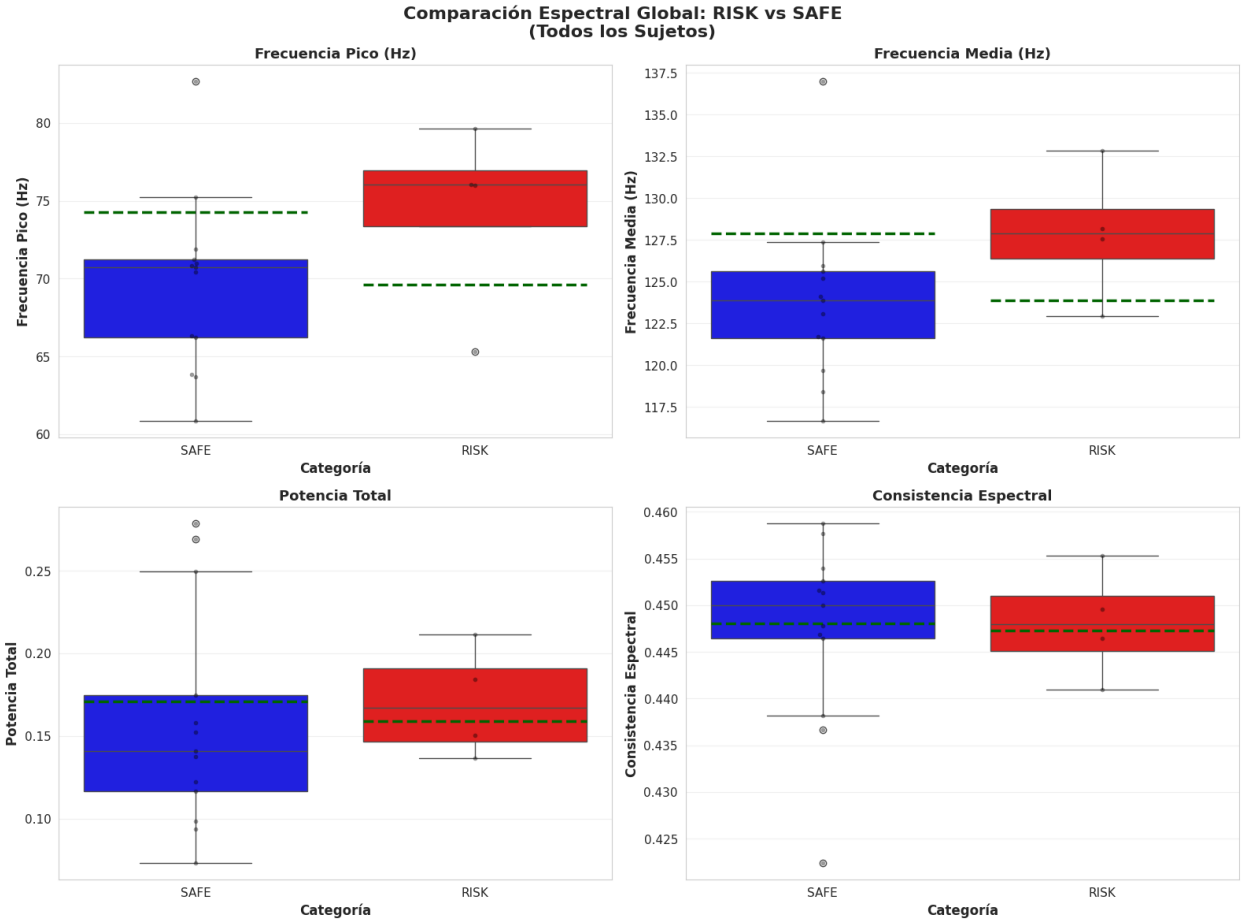


Figura 7.3: Comparación espectral global RISK vs SAFE

7.1.2. Funcionamiento del sistema

El sistema se organiza en una arquitectura modular compuesta por tres bloques principales. El primer componente corresponde al entorno de entrenamiento, implementado en Google Colab, donde se desarrolló y ejecutó el pipeline de procesamiento y modelado. Este módulo constituye la base de la arquitectura general del sistema, ya que genera los modelos y escaladores que posteriormente se integran al servidor Flask para su despliegue.

En este entorno se trabajó con la totalidad de los 40 sujetos, cada sujeto cuenta con un conjunto de variables que incluyen las señales sEMG y la información asociada a cada sesión de registro. En particular, cada archivo contiene 12 canales de sEMG con una longitud promedio de 1,808,331 muestras y una frecuencia de muestreo de 2000 Hz. Las demás variables del archivo incluyen el identificador del ejercicio (exercise), el identificador del sujeto (subject), las etiquetas de movimiento

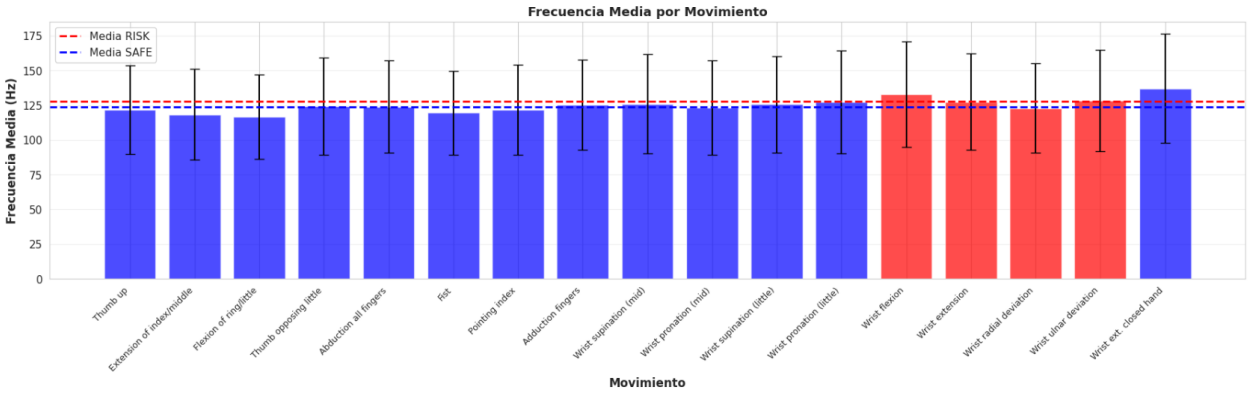


Figura 7.4: Frecuencia media por movimiento

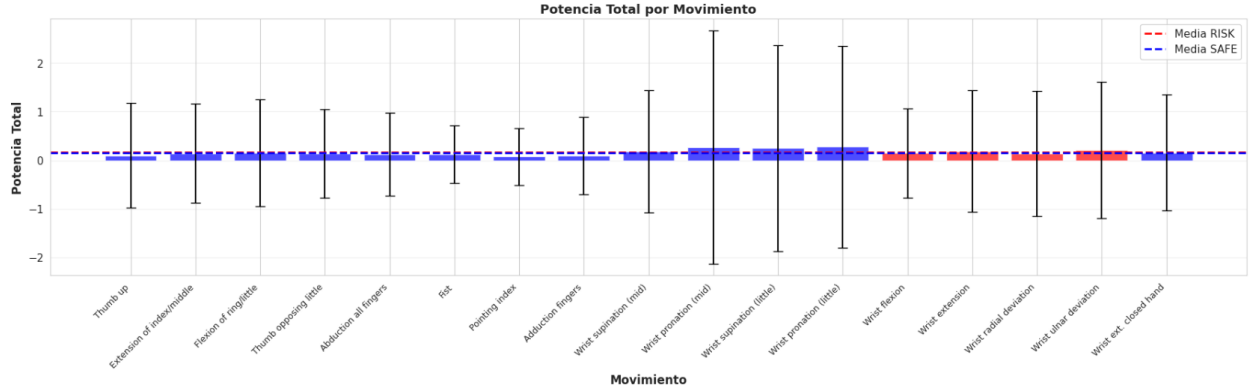


Figura 7.5: Potencia por movimiento

mostradas durante la adquisición (stimulus y restimulus), las repeticiones de cada gesto (rerepetition) y el género del participante (gender). Todas las variables numéricas se almacenan como enteros de 8 bits, excepto la señal sEMG, que se guarda como valores tipo single precision.

Cuadro 7.2: Estructura de los archivos .mat

Name	Value	Size	Class
emg	1808331 × 12 single	1808331 × 12	single
exercise	1	1 × 1	int8
frequency	2000	1 × 1	double
gender	'm/f'	1 × 1	char
rerepetition	1 × 1808331 int8	1 × 1808331	int8
restimulus	1 × 1808331 int8	1 × 1808331	int8
subject	1	1 × 1	int8

El preprocesamiento, como se observa en la figura 7.6, comprendió en primer lugar un filtrado pasabanda Butterworth de 20–450 Hz, rango en el cual se concentra el contenido de frecuencia relevante de las señales EMG de superficie [21]. Posteriormente, se aplicó un filtro Notch centrado en 50 Hz para eliminar el ruido de la red eléctrica, debido a que las señales fueron adquiridas con una red eléctrica a 50 Hz, como se especifica en la base de datos Ninapro.

El conjunto de datos fue dividido siguiendo un esquema estratificado por repeticiones de cada movimiento, asegurando independencia temporal entre entrenamiento (repeticiones 1, 3, 4, 6), validación (repetición 2) y prueba (repetición 5). Esta división permitió generar un total aproximado de 29,743 ventanas para entrenamiento, 7,429 para validación y 7,354 para prueba. Las clases fueron distribuidas de acuerdo con los movimientos de interés definidos como: Clase de riesgo (movimientos 13, 14, 15 y 16) y Clase segura (movimientos 1–12 y 17).

Los resultados del preprocesamiento mostraron una relación aproximada de 1:4 entre las clases de riesgo y seguras:

- **Train:** 5,861 ventanas de riesgo y 23,882 seguras.
- **Val:** 1,462 de riesgo y 5,967 seguras.
- **Test:** 1,415 de riesgo y 5,939 seguras.

Luego de la separación en los conjuntos se realizó una normalización tipo Z-score para estandarizar la amplitud de las señales. Las señales preprocesadas fueron segmentadas en ventanas de 1000 muestras (500 ms) con un solapamiento del 25 %. Las señales se procesaron mediante dos flujos paralelos según el tipo de modelo a entrenar, ver en la figura 7.7. Para DL se conservaron las secuencias crudas de señal en formato tridimensional (samples × time × channels). Para ML, se extrajeron 144 características por ventana en los dominios temporal, frecuencial y tiempo–frecuencia, incluyendo las siguientes caracteristicas:

- **Características en el dominio temporal:** cuantifican propiedades estadísticas y morfológicas de la señal en el tiempo, permitiendo describir su comportamiento muscular directo.

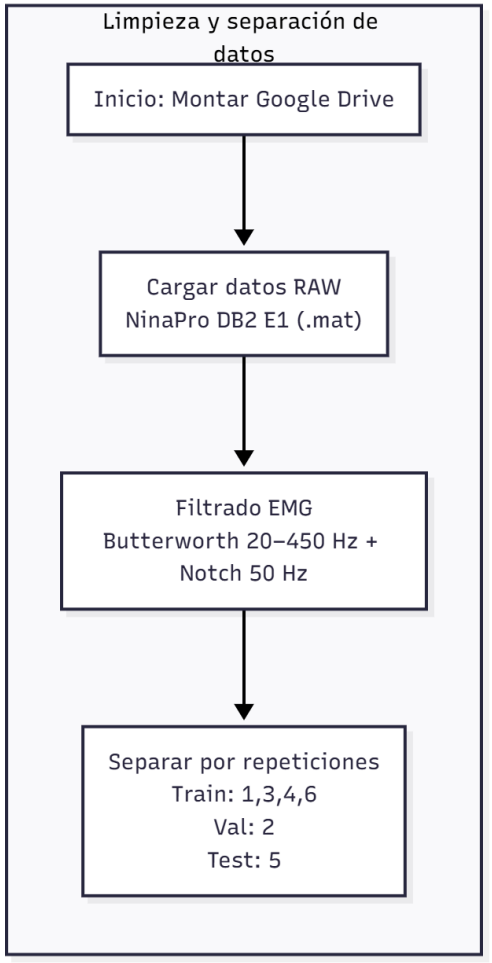


Figura 7.6: Limpieza y separación de datos

- **MAV (Mean Absolute Value):** valor medio absoluto de la señal, representa el nivel promedio de activación muscular en una ventana.
- **WL (Waveform Length):** longitud total de la forma de onda, refleja la complejidad o variabilidad del patrón electromiográfico.
- **ZC (Zero Crossing):** número de cruces por cero, indica la frecuencia de cambio de polaridad de la señal y está asociada con la frecuencia de disparo de las unidades motoras.
- **SSC (Slope Sign Changes):** número de cambios de signo en la pendiente de la señal, mide la variación de la forma de onda y la complejidad temporal.
- **RMS (Root Mean Square):** raíz cuadrática media, relacionada con la potencia de la señal y con el nivel de contracción muscular.

- **Características en el dominio frecuencial:** describen la distribución de energía espectral

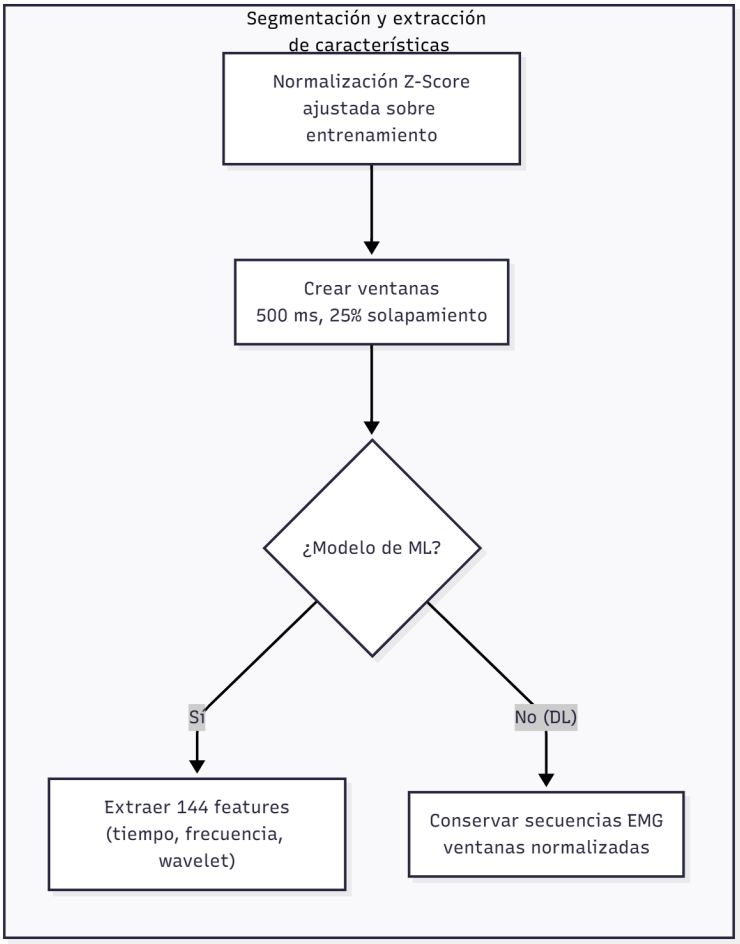


Figura 7.7: Segmentación y extracción de características

de la señal y permiten identificar patrones asociados a fatiga o tipos de activación muscular.

- **Mean Frequency ($f_{\mathbf{mean}}$):** promedio ponderado de las frecuencias presentes en el espectro de potencia.
- **Median Frequency ($f_{\mathbf{med}}$):** frecuencia que divide el espectro de potencia en dos áreas de energía iguales, utilizada comúnmente como indicador de fatiga muscular.

- **Características en el dominio tiempo–frecuencia (Wavelet):** obtenidas mediante la **Transformada Discreta Wavelet (DWT)** usando la familia *Daubechies 4* (`db4`) con cuatro niveles de descomposición.

  - **Energía por nivel:** para cada subbanda wavelet (c_j) Esta métrica cuantifica la distribución de energía de la señal en diferentes bandas de frecuencia y permite caracterizar fenómenos transitorios y variaciones locales del patrón EMG.

Dependiendo del experimento, se aplicó la técnica de balanceo correspondiente: SMOTE o ADASYN para los modelos ML, y Data Augmentation y/o Focal Loss para los modelos DL, tal como se muestra en la figura 7.8.

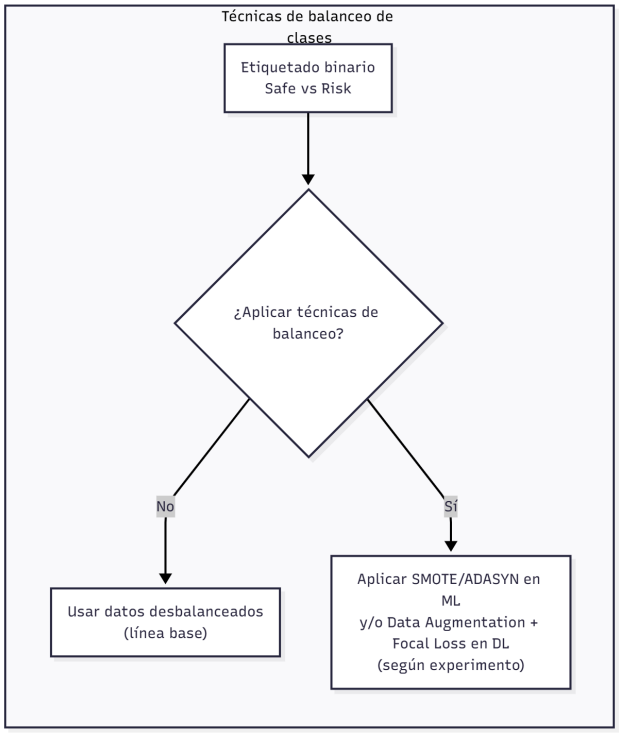


Figura 7.8: Técnicas de balanceo de clases

Posteriormente, se entrenaron cuatro modelos elegidos con base en la revisión bibliográfica y en el rendimiento observado durante el prototipado inicial: los modelos de Machine Learning (Ensemble Subspace KNN y Random Forest) construidos sobre las 144 características extraídas por ventana, y los modelos de Deep Learning (CNN+LSTM con atención y Bidirectional LSTM con atención) que operan directamente sobre las secuencias EMG preprocesadas. Dependiendo de la configuración seleccionada para el experimento se realizó la búsqueda de hiperparametros (Random Search) o se continuó el entrenamiento y testeo de los modelos, ver la figura 7.9.

Una vez completada la fase de entrenamiento, los modelos obtenidos fueron exportados en los formatos correspondientes: archivos .pkl para los modelos ML y archivos .keras para los modelos DL. Adicionalmente, se generó el escalador global (scaler.pkl) y la metadata necesaria para reproducir el proceso de inferencia. Todos estos artefactos fueron posteriormente transferidos al servidor local, donde quedaron integrados en la etapa de despliegue junto con la API Flask que ejecuta el pipeline de clasificación.

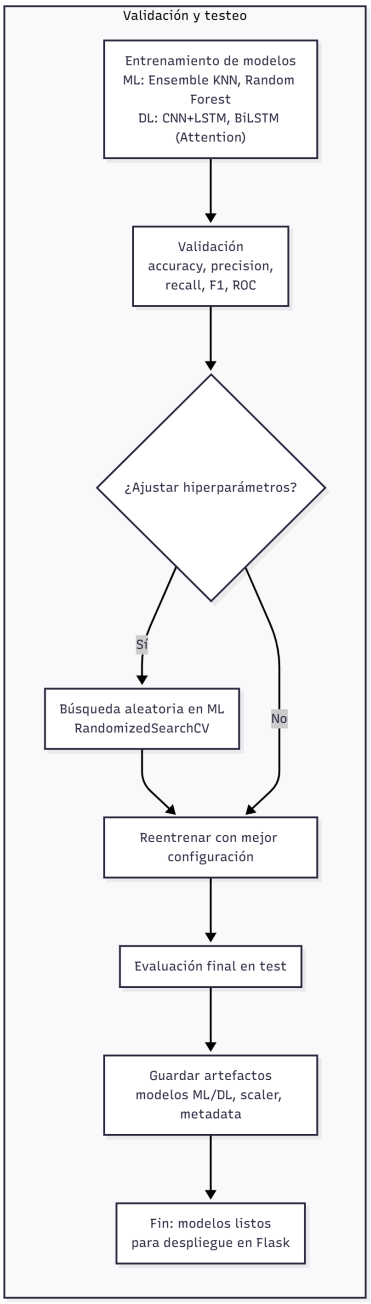


Figura 7.9: Validación y testeo

7.2. Evaluación de modelos de aprendizaje automático

Durante la fase de reconocimiento de los datos se identificó desbalanceo entre las clases del conjunto de datos, con una cantidad mayor de ventanas pertenecientes a la clase segura en comparación con la clase de riesgo (relación 1:4). Esta asimetría en la distribución de clases puede afectar de manera directa la capacidad de generalización de los modelos, sesgando el entrenamiento hacia la clase dominante y reduciendo la sensibilidad en la detección de eventos poco frecuentes.

Con el objetivo de mitigar este problema y analizar su impacto sobre el desempeño de los modelos, se implementaron y compararon estrategias de balanceo de clases:

Con el objetivo de mitigar el desbalance entre las clases y evaluar su impacto sobre el desempeño de los modelos, se implementaron y compararon diversas estrategias de balanceo de datos:

- **SMOTE (Synthetic Minority Over-sampling Technique):** técnica de sobremuestreo que genera instancias sintéticas de la clase minoritaria mediante interpolación con sus vecinos más cercanos en el espacio de características. Se utilizó exclusivamente en los modelos de Machine Learning.

- **ADASYN (Adaptive Synthetic Sampling):** variante de SMOTE que ajusta dinámicamente la cantidad de muestras sintéticas generadas en función de la dificultad local de aprendizaje. En este enfoque, las regiones donde la clase minoritaria es más escasa reciben mayor densidad de muestras artificiales, favoreciendo una frontera de decisión más equilibrada.

- **Data Augmentation para Deep Learning:** conjunto de transformaciones aplicadas sobre las ventanas de señal EMG, incluyendo *jittering* (ruido aditivo), *scaling* (cambios globales de amplitud) y *time warping* (deformaciones temporales suaves). Esta técnica incrementa la diversidad del conjunto de entrenamiento sin alterar la naturaleza fisiológica de la señal.

- **Focal Loss:** función de pérdida utilizada en los modelos de Deep Learning para mitigar el impacto del desbalance sin generar nuevas muestras. Modula la pérdida mediante los parámetros γ y α, potenciando la contribución de ejemplos difíciles y otorgando mayor peso a la clase minoritaria.

Las técnicas de balanceo fueron integradas dentro del pipeline de entrenamiento implementado en Google Colab, lo que permitió realizar variaciones controladas de la estrategia aplicada en cada experimento y evaluar su impacto directo sobre las métricas de desempeño. Con este fin, se establecieron cinco escenarios experimentales: entrenamiento sin balanceo, balanceo mediante SMOTE para modelos de ML combinado con data augmentation y Focal Loss en los modelos de DL, balanceo mediante ADASYN para ML junto con data augmentation y Focal Loss en DL, ADASYN para ML combinado únicamente con data augmentation en DL, y ADASYN para ML junto con Focal Loss en DL sin data augmentation.

Inicialmente se realizó una corrida exploratoria con el conjunto de datos sin aplicar técnicas de balanceo de clases, con el propósito de establecer una línea base de desempeño para los modelos clásicos de machine learning. En esta etapa se habilitó un proceso de búsqueda aleatoria de hiperparámetros (Random Search) con el fin de identificar las combinaciones óptimas de los parámetros de entrenamiento para cada modelo, a partir de los espacios definidos en el código experimental.

Para el modelo Ensemble Subspace KNN, la búsqueda de hiperparámetros mediante Random Search evaluó combinaciones de n_estimators, max_samples, max_features y n_neighbors. El mejor desempeño se obtuvo con la configuración: 30 estimadores, max_samples = 0.7, max_features = 0.8 y k = 7.

En el caso del Random Forest, Random Search exploró variaciones en n_estimators, max_depth, min_samples_split y min_samples_leaf. El modelo óptimo correspondió a la configuración con 200 árboles, profundidad máxima de 30, min_samples_split = 5, min_samples_leaf = 2 y selección de características mediante la estrategia sqrt.

7.2.1. Experimento 1 — Sin balanceo

Este primer experimento establece la línea base del sistema, entrenando todos los modelos sin aplicar técnicas de balanceo. La Figura 7.10 confirma una marcada desproporción entre las clases Safe (23,882 muestras) y Risk (5,861 muestras), lo que genera un conjunto de entrenamiento sesgado hacia la clase mayoritaria. Este escenario permite evaluar el comportamiento natural de los modelos ante un desbalance real, sin alteraciones en los datos.

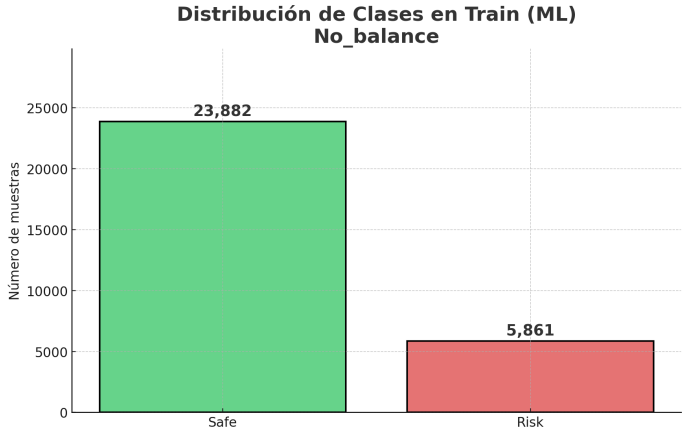


Figura 7.10: Distribución de clases en el conjunto de entrenamiento sin balanceo.

La Figura 7.11 presenta las curvas ROC en el conjunto de prueba. Los modelos de Machine Learning obtuvieron los mayores valores de AUC, destacando Random Forest con 0.785 y el Ensemble

KNN con 0.734. Entre los modelos profundos, CNN–LSTM con atención alcanzó un AUC de 0.773, seguido del BiLSTM con 0.740. En general, todos los modelos muestran una capacidad moderada de discriminación, limitada por el fuerte desbalance.

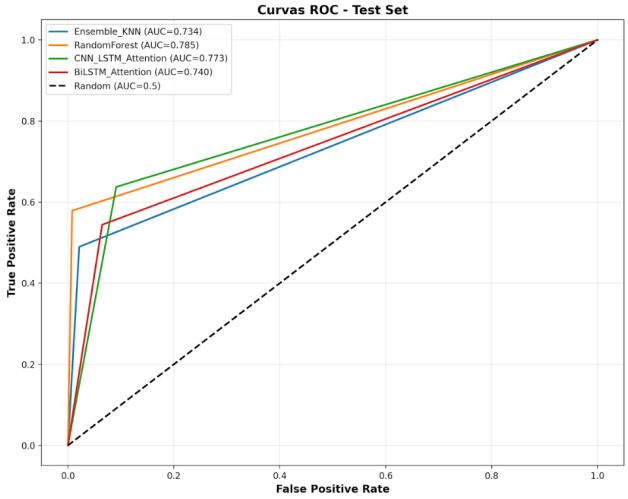


Figura 7.11: Curvas ROC de los modelos en el conjunto de prueba (sin balanceo).

Las matrices de confusión (Figuras 7.12 y 7.13) evidencian cómo el desbalance afecta la sensibilidad hacia la clase Risk.

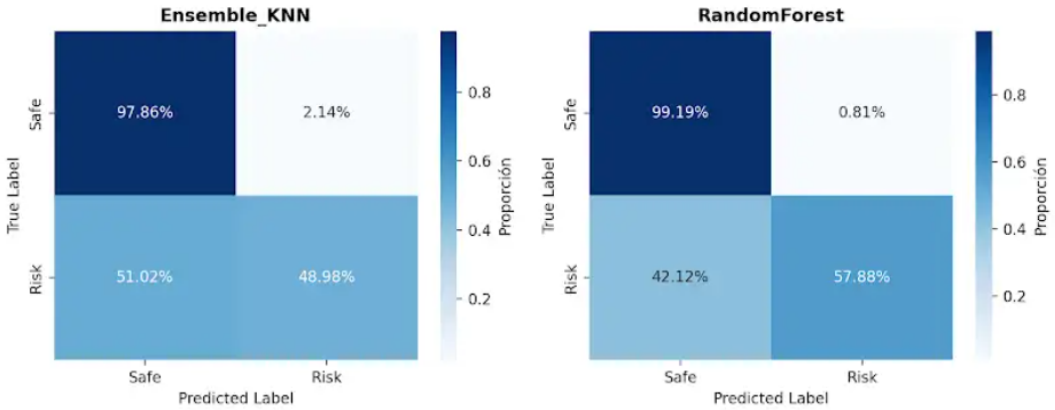


Figura 7.12: Matrices de confusión normalizadas para Ensemble KNN y Random Forest.

Random Forest clasifica correctamente el 99.19 % de las muestras Safe, pero solo el 57.88 % de las muestras Risk. Aunque este desempeño es mejor que lo esperado en escenarios de desbalance severo, sigue evidenciando una fuerte inclinación hacia la clase mayoritaria.

El Ensemble KNN presenta un 48.98 % de acierto en la clase Risk y 97.86 % en Safe. Aunque su sensibilidad es inferior a la de Random Forest.

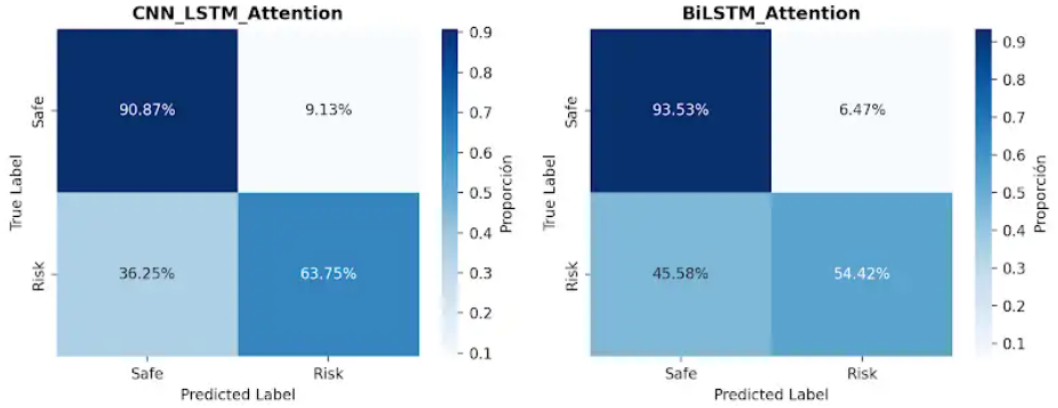


Figura 7.13: Matrices de confusión normalizadas para CNN_LSTM_Attention y BiLSTM_Attention.

En los modelos de Deep Learning, la arquitectura CNN-LSTM con atención logra la mayor sensibilidad entre los modelos profundos, identificando correctamente el 63.75 % de las muestras Risk. El modelo BiLSTM alcanza un 54.42 % en esta clase, inferior al CNN-LSTM.

Las curvas de aprendizaje de ambos modelos profundos se muestran en las Figuras 7.14 y 7.15. La CNN-LSTM presenta una convergencia estable, sin señales marcadas de sobreajuste, mientras que el BiLSTM exhibe mayor variabilidad en la pérdida de validación.

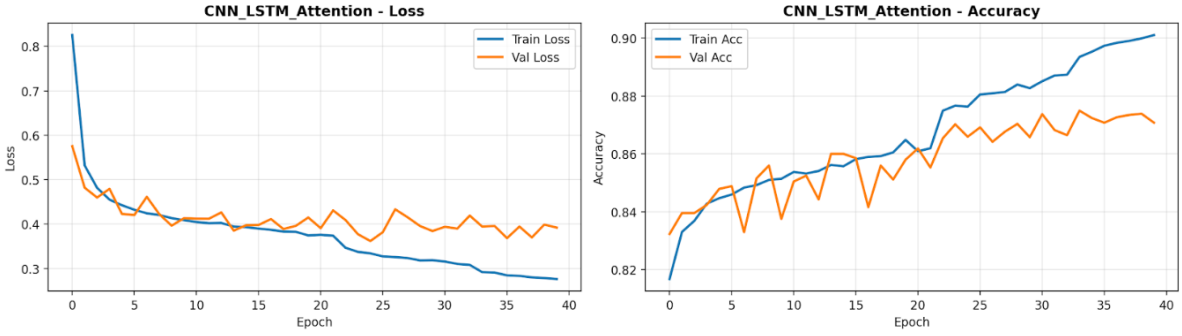


Figura 7.14: Curvas de pérdida y precisión para el modelo CNN-LSTM con atención.

La Figura 7.16 resume el desempeño global. Random Forest obtiene la precisión más alta (0.945), aunque con un recall moderado (0.579). El mayor F1-score corresponde a CNN-LSTM con 0.631,

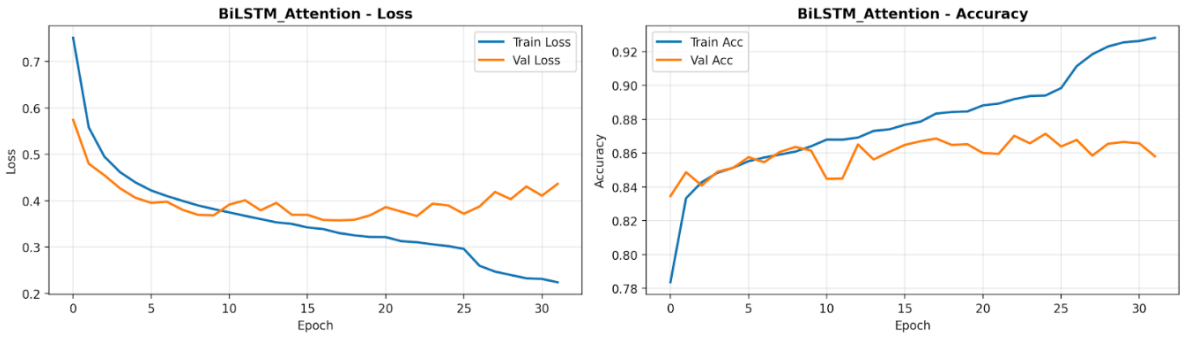


Figura 7.15: Curvas de pérdida y precisión para el modelo BiLSTM con atención.

seguido por el Ensemble KNN con 0.620. Los modelos profundos muestran mejores resultados en la detección de la clase minoritaria, mientras que los modelos clásicos alcanzan mayor precisión gracias al sesgo inducido por el desbalance.

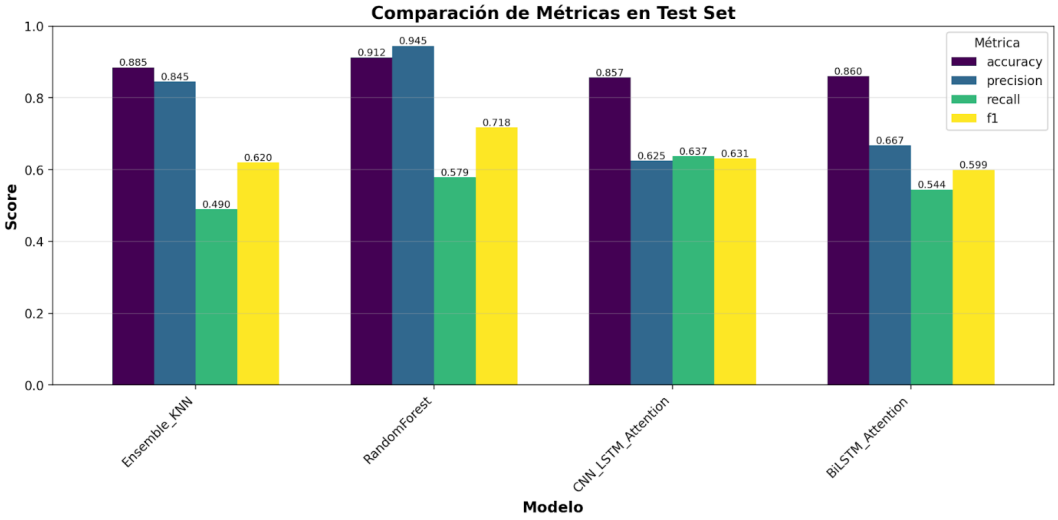


Figura 7.16: Comparación de métricas en el conjunto de prueba sin balanceo.

7.2.2. Experimento 2 — SMOTE para ML y Data Augmentation con Focal Loss para DL

En este segundo experimento se aplicaron técnicas de balanceo diferenciadas para cada tipo de modelo. En los algoritmos de Machine Learning se utilizó SMOTE, cuyo objetivo es generar muestras sintéticas de la clase minoritaria hasta igualar la cantidad de ejemplos de la clase mayoritaria. Para los modelos de Deep Learning se empleó una combinación de Focal Loss (gamma = 2.5, alpha

= 0.75) y técnicas de data augmentation en las ventanas crudas de EMG, incluyendo jittering, scaling y time warping. Estas estrategias buscan reducir el sesgo hacia la clase Safe y mejorar la capacidad de detección de la clase Risk.

La Figura 7.17 muestra el efecto de SMOTE sobre el conjunto de entrenamiento de Machine Learning. Después del balanceo, ambas clases quedan equilibradas en 23,882 muestras, eliminando por completo la desproporción presente en el experimento anterior.

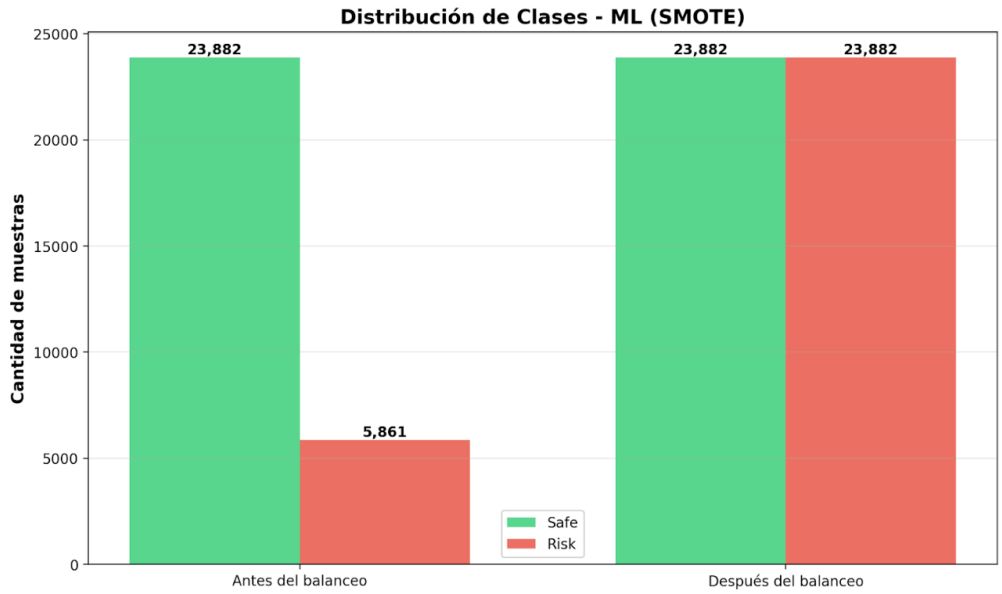


Figura 7.17: Distribución de clases antes y después del balanceo con SMOTE para los modelos de Machine Learning.

La Figura 7.18 presenta las curvas ROC en el conjunto de prueba. El balanceo tuvo un impacto notable en los modelos de Machine Learning: Random Forest alcanzó el mayor AUC con un valor de 0.900, seguido del Ensemble KNN con 0.831. En los modelos de Deep Learning se observa un comportamiento más moderado, con AUC de 0.700 tanto para CNN-LSTM con atención como para BiLSTM, aunque con una forma de curva más conservadora que en el experimento previo debido al uso de Focal Loss.

Las matrices de confusión de los modelos de Machine Learning se presentan en la Figura 7.19. El Ensemble KNN logra identificar correctamente el 86.29 % de las muestras Risk, una mejora notable respecto al experimento previo. Random Forest alcanza un 83.96 % en la clase Risk, manteniendo al mismo tiempo un 95.99 % de aciertos en la clase Safe, lo que refleja un equilibrio sólido gracias al balanceo aplicado.

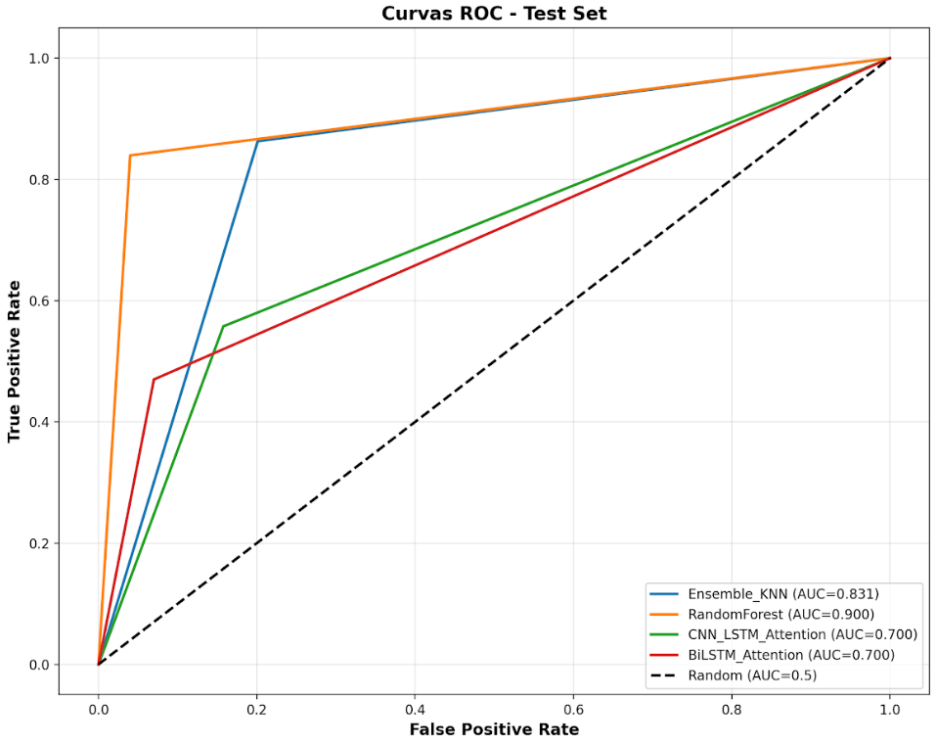


Figura 7.18: Curvas ROC de los modelos en el conjunto de prueba para el experimento con SMOTE y Focal Loss.

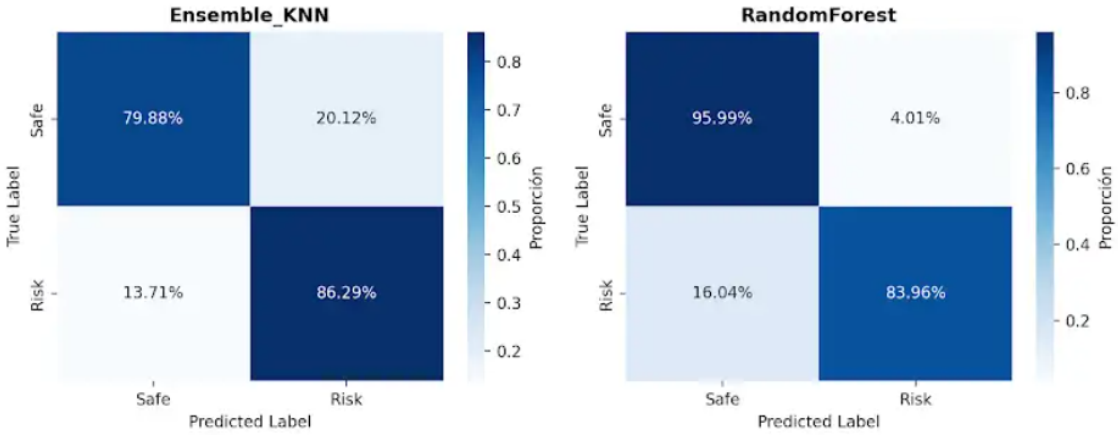


Figura 7.19: Matrices de confusión normalizadas para Ensemble KNN y Random Forest después de aplicar SMOTE.

En los modelos de Deep Learning (Figura 7.20), la CNN-LSTM con atención alcanza un 55.76 % de acierto en la clase Risk, mientras que el BiLSTM obtiene un 47 %. Aunque los valores son in-

feriores a los alcanzados por los modelos clásicos balanceados, muestran una mejora respecto al experimento previo en la arquitectura CNN-LSTM, atribuible al uso de Focal Loss.

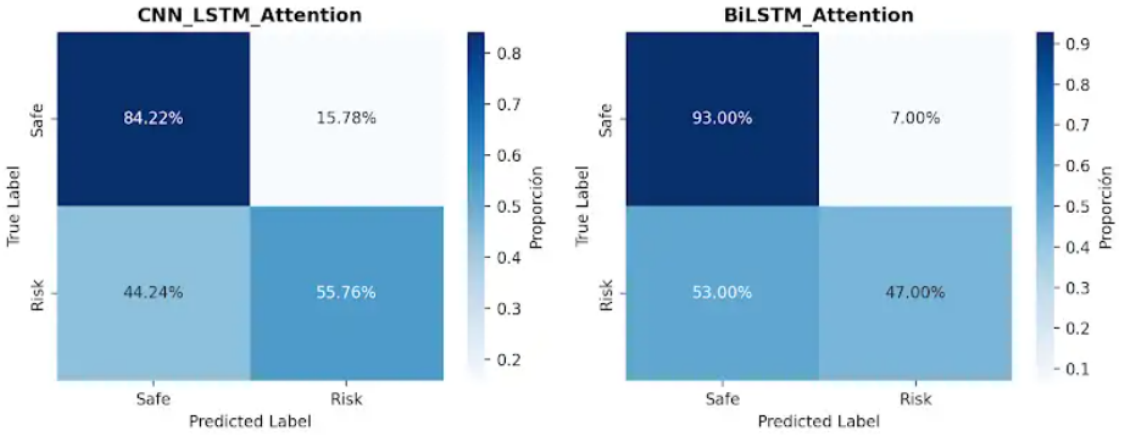


Figura 7.20: Matrices de confusión normalizadas para CNN-LSTM con atención y BiLSTM con atención.

Las curvas de entrenamiento (Figuras 7.21 y 7.22) muestran una convergencia más estable que en el experimento anterior. Tanto CNN-LSTM como BiLSTM presentan pérdidas de validación relativamente bajas y comportamientos consistentes entre entrenamiento y validación, lo que indica que las técnicas de augmentación contribuyeron a mejorar la generalización.

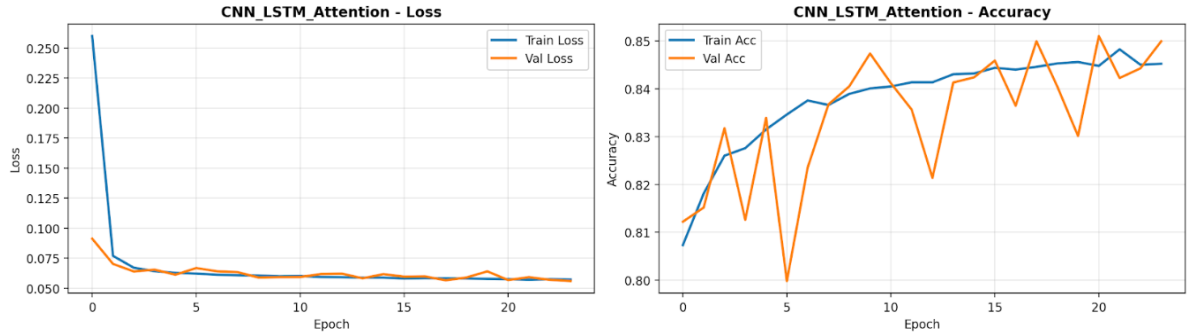


Figura 7.21: Curvas de pérdida y precisión para el modelo CNN-LSTM con atención utilizando Focal Loss y data augmentation.

La Figura 7.23 resume los valores de exactitud, precisión, recall y F1-score para los cuatro modelos. Random Forest obtiene el mejor desempeño global con un F1-score de 0.836 y valores equilibrados de precisión y recall. El Ensemble KNN también muestra un avance considerable gracias al balanceo, alcanzando un recall de 0.863 y un F1-score de 0.637. En comparación, los modelos

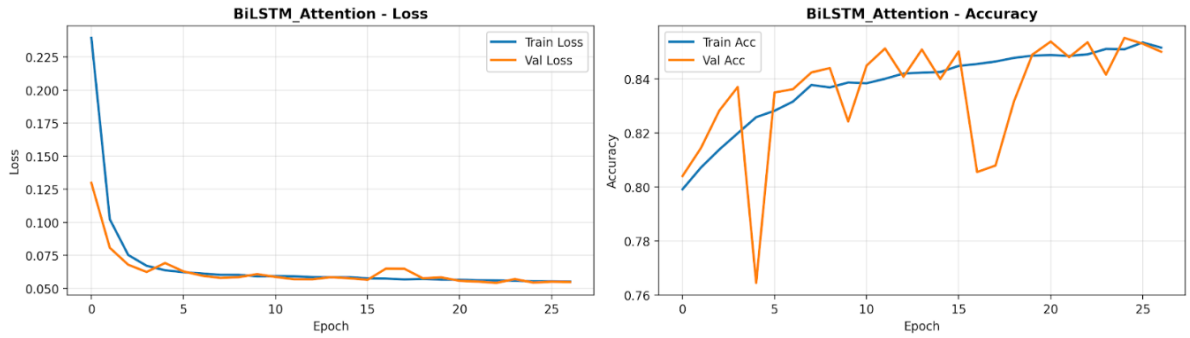


Figura 7.22: Curvas de pérdida y precisión para el modelo BiLSTM con atención utilizando Focal Loss y data augmentation.

de Deep Learning presentan métricas más discretas, aunque con mejoras en estabilidad y recall respecto al experimento anterior.

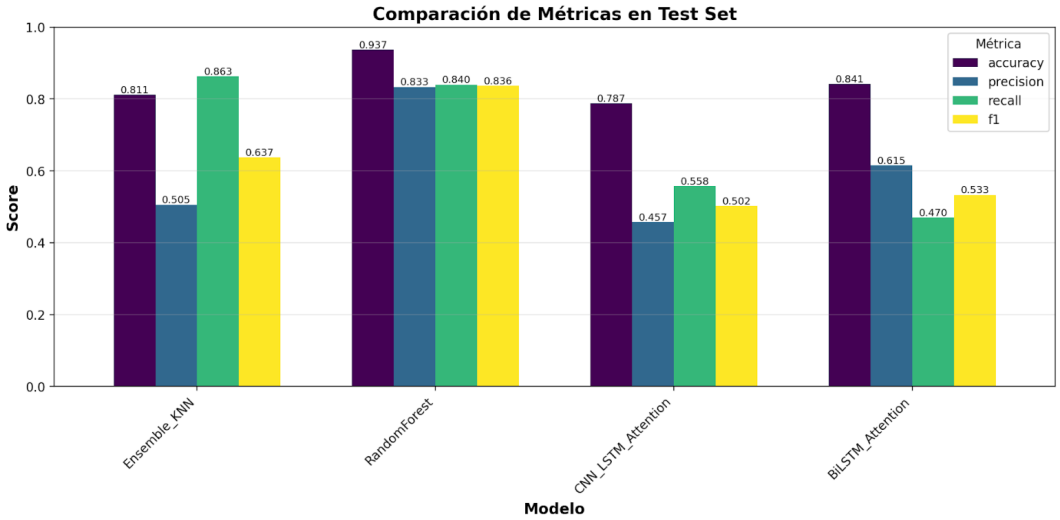


Figura 7.23: Comparación de métricas en el conjunto de prueba para el experimento con SMOTE, Focal Loss y data augmentation.

7.2.3. Experimento 3 — ADASYN para ML y Focal Loss con Data Augmentation para DL

En este tercer experimento se aplicó ADASYN como técnica de balanceo para los modelos de Machine Learning. A diferencia de SMOTE, ADASYN genera más instancias sintéticas en las regiones de mayor dificultad de clasificación, aumentando la densidad de la clase minoritaria en

zonas donde el modelo presenta más incertidumbre. Para los modelos de Deep Learning se mantuvo la misma estrategia del experimento anterior: combinación de Focal Loss con parámetros gamma = 2.5 y alpha = 0.75, junto con técnicas de data augmentation aplicadas sobre las ventanas de EMG.

La Figura 7.24 muestra el efecto del balanceo generado por ADASYN. Después del proceso, la clase Risk aumenta de 5,861 a 23,983 muestras, logrando un balance casi perfecto frente a la clase Safe.

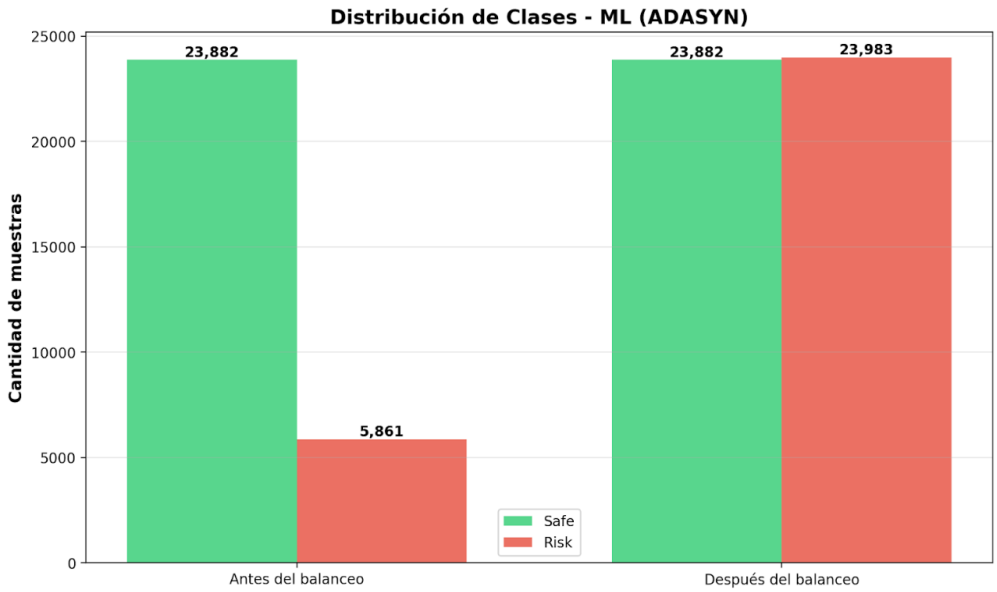


Figura 7.24: Distribución de clases antes y después del balanceo con ADASYN para los modelos de Machine Learning.

La Figura 7.25 presenta las curvas ROC obtenidas en el conjunto de prueba. El modelo Random Forest obtiene el mayor AUC del experimento con un valor de 0.906, seguido por el Ensemble KNN con 0.828. Los modelos profundos alcanzan valores de AUC de 0.681 para el CNN-LSTM con atención y 0.706 para el BiLSTM con atención. En general, ADASYN favorece principalmente a los modelos clásicos, aunque los modelos de Deep Learning mantienen un desempeño estable y similar al observado en el experimento anterior.

Las matrices de confusión de los modelos de Machine Learning (Figura 7.26) muestran un avance en la capacidad de detección de la clase Risk. El Ensemble KNN alcanza un 90.25 % en dicha clase, y Random Forest obtiene un 86.08 %, ademas, en la clase Safe obtienen valores de 75.32 % y 95.12 % respectivamente.

En el caso de los modelos profundos (Figura 7.27), el CNN-LSTM con atención clasifica correc-

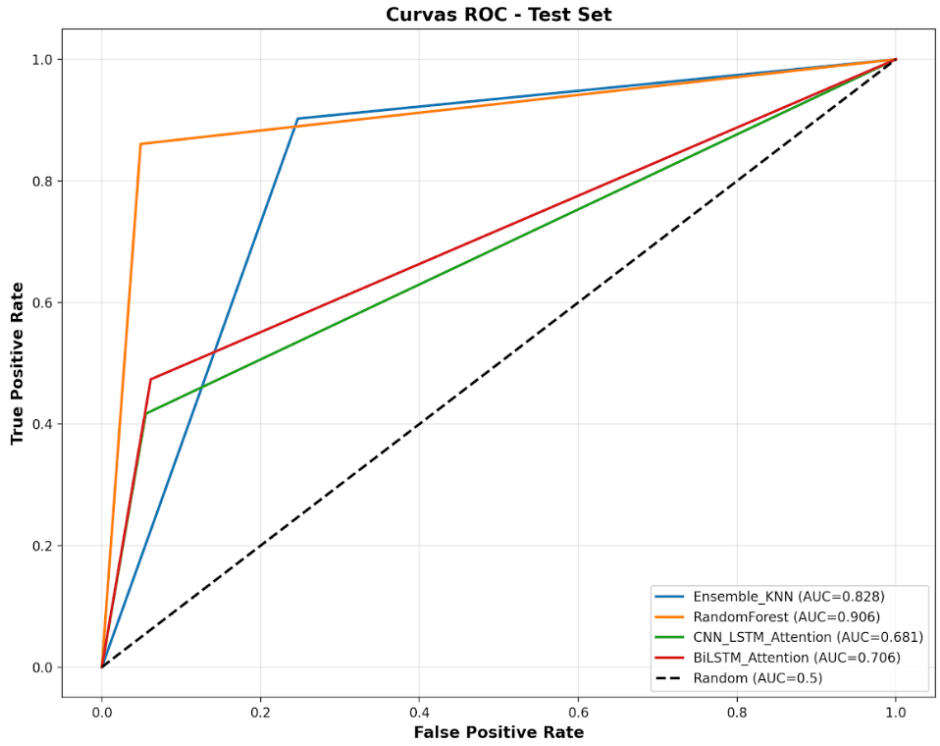


Figura 7.25: Curvas ROC de los modelos en el conjunto de prueba para el experimento con ADASYN y Focal Loss.

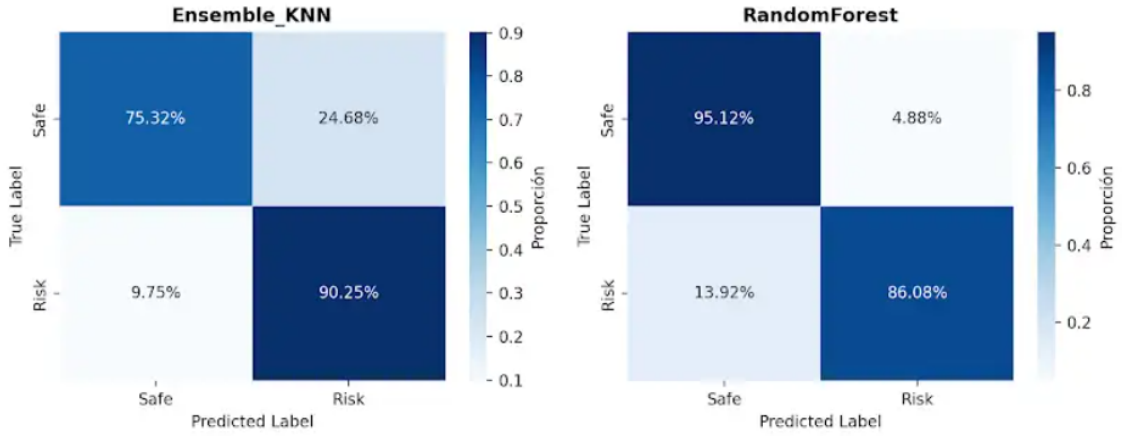


Figura 7.26: Matrices de confusión normalizadas para Ensemble KNN y Random Forest después de aplicar ADASYN.

tamente el 41.70 % de las muestras Risk, mientras que el BiLSTM con atención alcanza el 47.35 %. Aunque estos valores no superan a los modelos de Machine Learning balanceados, ambos modelos

mantienen un nivel de estabilidad durante la predicción, coherente con los valores de AUC.

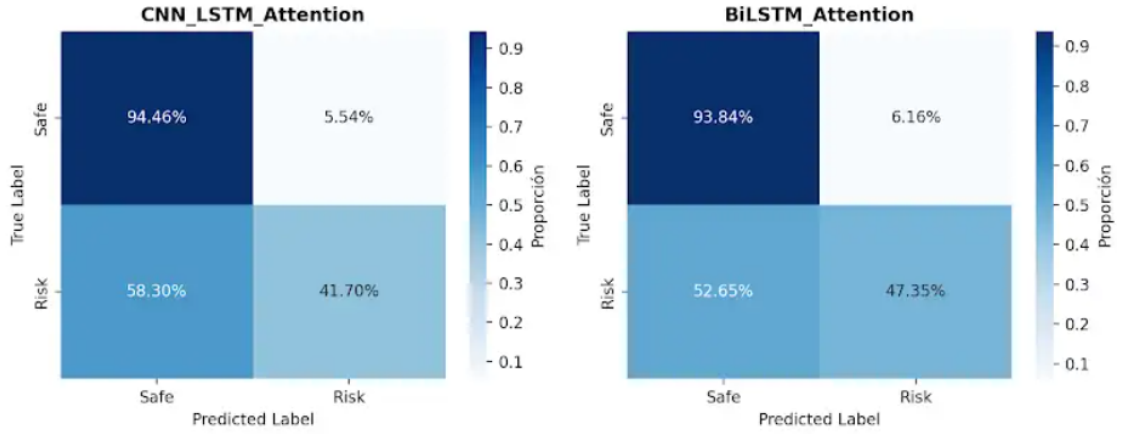


Figura 7.27: Matrices de confusión normalizadas para CNN-LSTM con atención y BiLSTM con atención.

Las curvas de entrenamiento de los modelos profundos se presentan en las Figuras 7.28 y 7.29. Ambos modelos muestran un comportamiento estable, con pérdidas de validación bajas y una convergencia progresiva en la precisión. El uso de Focal Loss continúa ayudando a evitar un sesgo excesivo hacia la clase Safe.

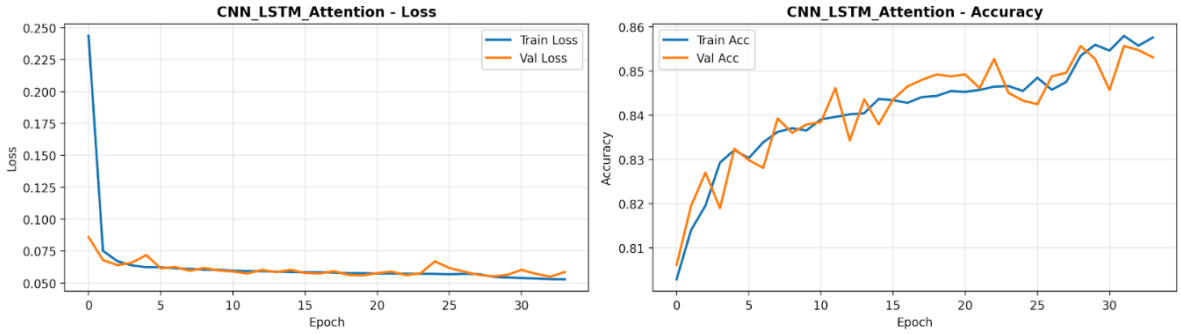


Figura 7.28: Curvas de pérdida y precisión para el modelo CNN-LSTM con atención en el experimento con ADASYN.

La Figura 7.30 resume los valores finales de exactitud, precisión, recall y F1-score. El mejor desempeño general vuelve a corresponder al modelo Random Forest, con un F1-score de 0.833 y métricas equilibradas entre precisión y recall. El Ensemble KNN mantiene un recall muy alto de 0.902, aunque a costa de una precisión moderada. En los modelos profundos, el BiLSTM obtiene el mejor F1-score con un valor de 0.547, superando al CNN-LSTM en esta métrica.

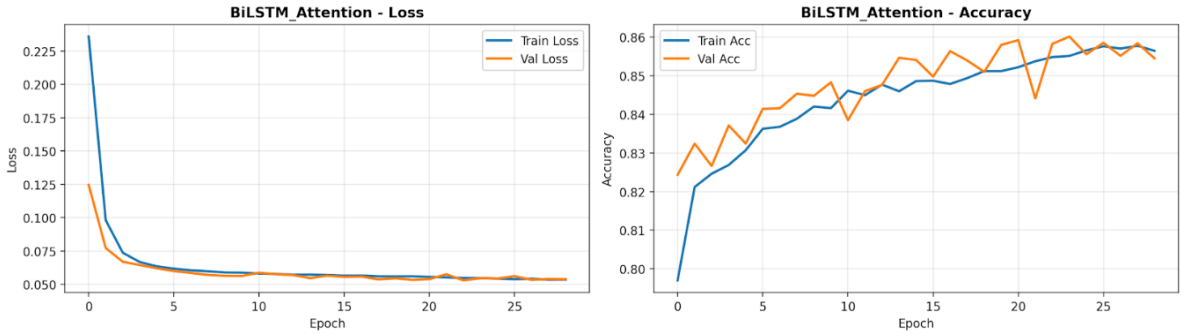


Figura 7.29: Curvas de pérdida y precisión para el modelo BiLSTM con atención en el experimento con ADASYN.

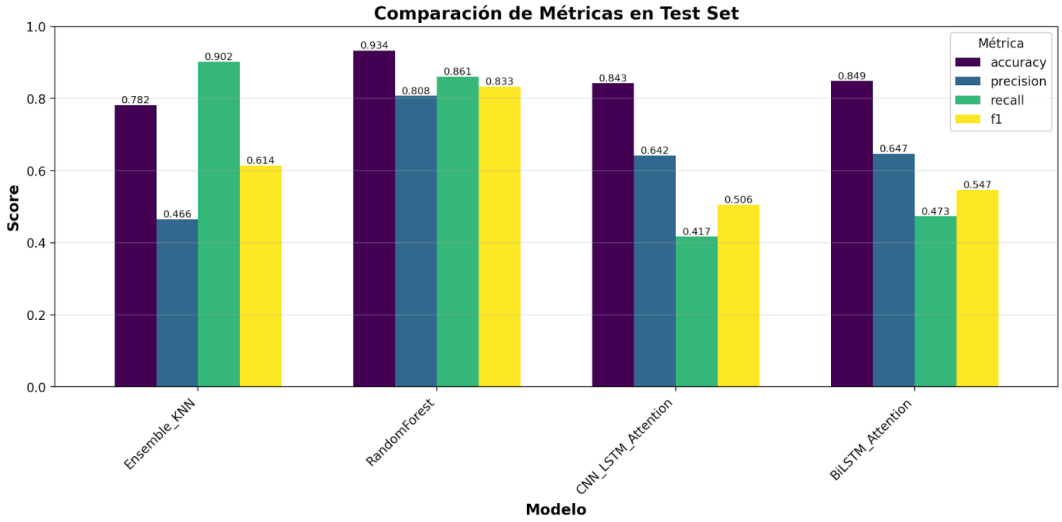


Figura 7.30: Comparación de métricas en el conjunto de prueba para el experimento con ADASYN, Focal Loss y data augmentation.

7.2.4. Experimento 4 — ADASYN para ML y Data Augmentation para DL

En este cuarto experimento se aplicó nuevamente ADASYN como técnica de balanceo para los modelos de Machine Learning, con el fin de incrementar la representatividad de la clase Risk en zonas donde el clasificador presenta mayor incertidumbre. En los modelos de Deep Learning se utilizó únicamente data augmentation sobre las ventanas de EMG, manteniendo la función de pérdida estándar Binary Crossentropy sin emplear Focal Loss. Esta configuración permite evaluar el efecto aislado de la augmentación en comparación con el experimento anterior.

La Figura 7.31 muestra la distribución de clases antes y después de aplicar ADASYN. La clase Risk aumenta hasta 23,983 muestras, obteniendo un balance prácticamente idéntico con la clase Safe.

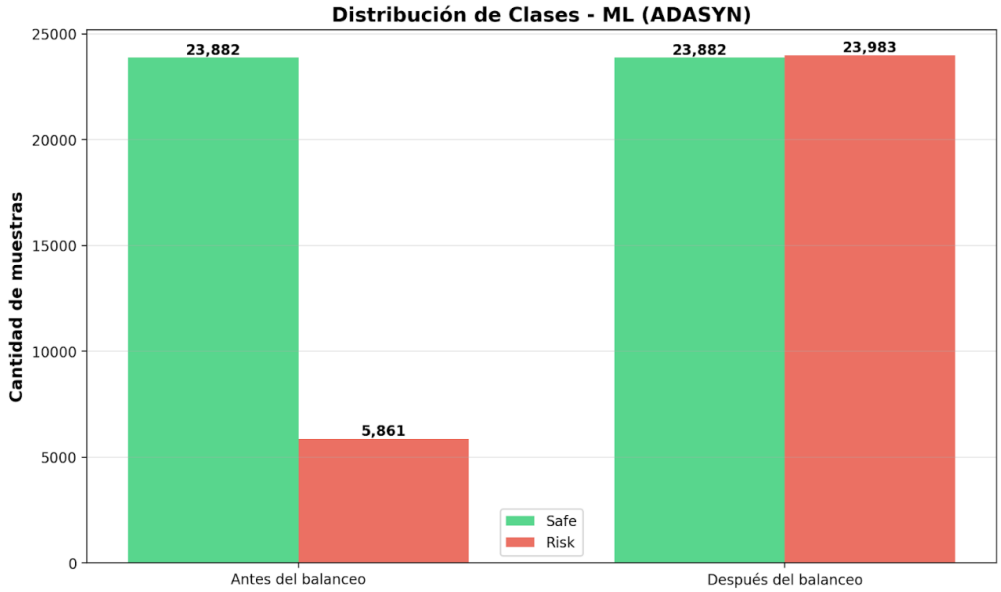


Figura 7.31: Distribución de clases antes y después del balanceo con ADASYN para los modelos de Machine Learning.

Las curvas ROC en el conjunto de prueba se presentan en la Figura 7.32. El modelo Random Forest mantiene el mejor desempeño global con un AUC de 0.906, seguido por el Ensemble KNN con un AUC de 0.828. En los modelos de Deep Learning, la arquitectura CNN-LSTM con atención alcanza un AUC de 0.764, mientras que el BiLSTM obtiene un valor de 0.760. Estos resultados muestran que, aunque no se utilizó Focal Loss, el data augmentation favorece la capacidad discriminativa de los modelos profundos en comparación con los experimentos previos.

Las matrices de confusión de los modelos de Machine Learning se muestran en la Figura 7.33. El Ensemble KNN logra un 90.25 % de acierto en la clase Risk, mientras que Random Forest alcanza un 86.08 %. En la clase Safe, los valores obtenidos son 75.32 % y 95.12 % respectivamente.

En los modelos de Deep Learning (Figura 7.34), el CNN-LSTM con atención obtiene un 61.06 % de acierto en la clase Risk y un 91.68 % en la clase Safe. El modelo BiLSTM alcanza un 64.52 % de acierto en la clase Risk y 87.42 % en la clase Safe. Estos valores representan una mejora respecto al experimento anterior, lo que sugiere que el uso aislado de data augmentation contribuye a mejorar la capacidad del modelo para reconocer la clase minoritaria sin la penalización adicional que introduce Focal Loss.

Las curvas de entrenamiento se presentan en las Figuras 7.35 y 7.36. Ambas arquitecturas muestran una convergencia progresiva y estable, con coherencia entre las curvas de entrenamiento y validación. En comparación con el experimento anterior, los modelos profundos presentan menor

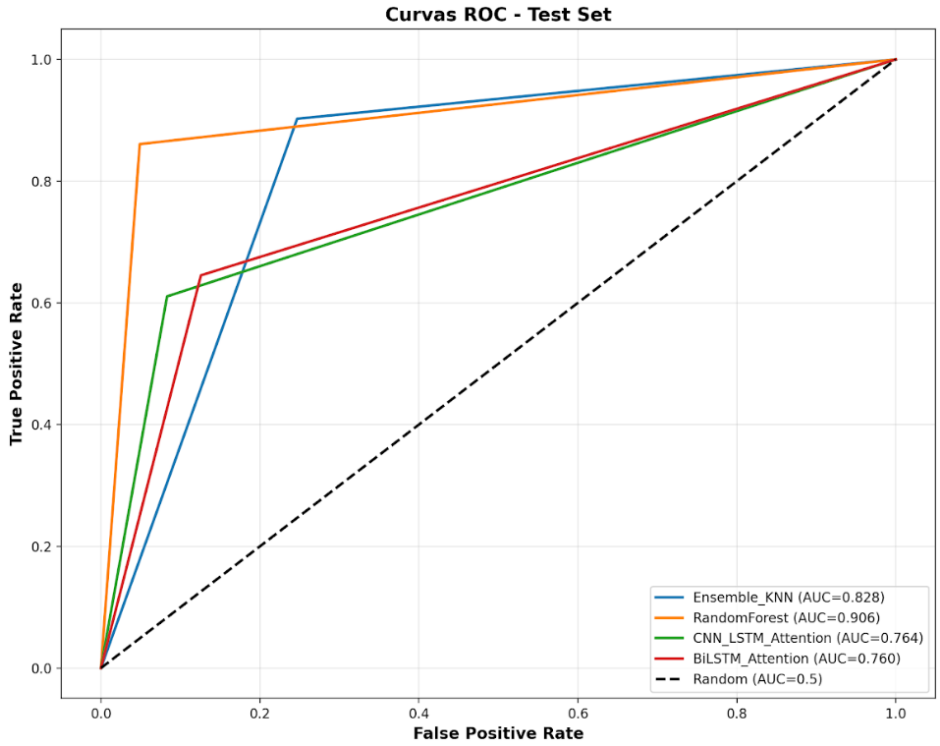


Figura 7.32: Curvas ROC de los modelos en el conjunto de prueba para el experimento con ADASYN y data augmentation.

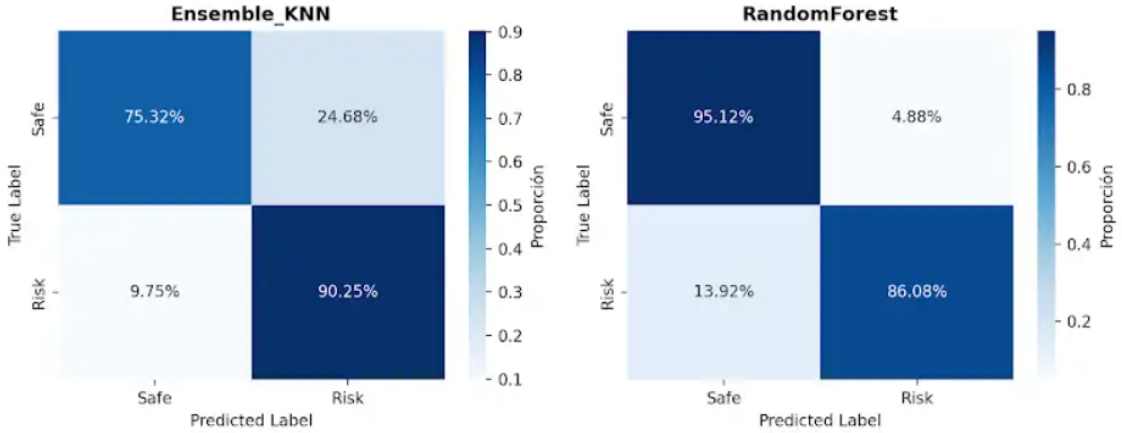


Figura 7.33: Matrices de confusión normalizadas para Ensemble KNN y Random Forest después de aplicar ADASYN.

variabilidad y muestran un aprendizaje más consistente.

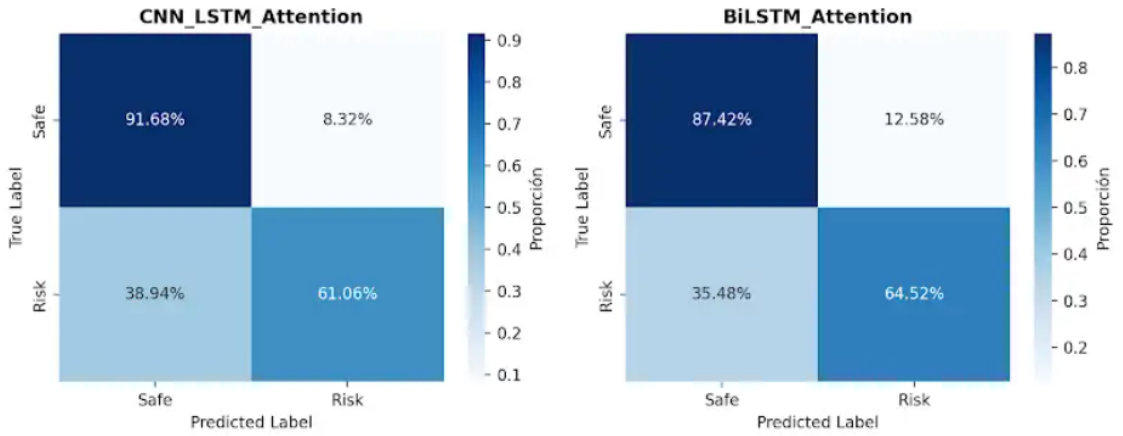


Figura 7.34: Matrices de confusión normalizadas para CNN-LSTM con atención y BiLSTM con atención.

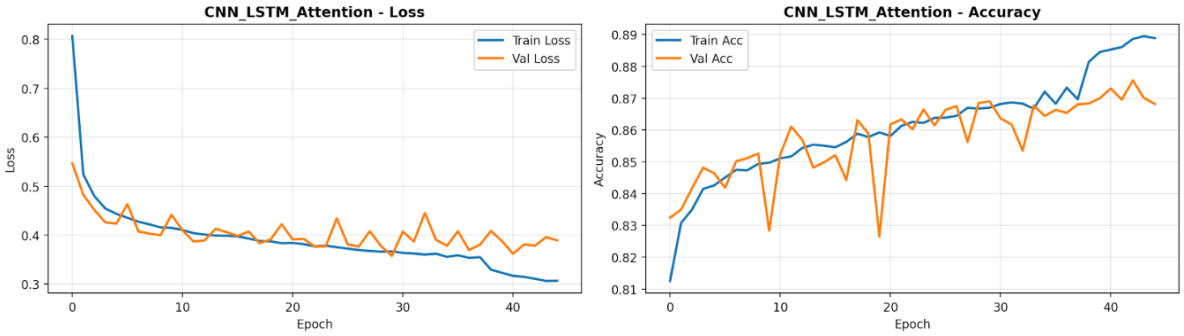


Figura 7.35: Curvas de pérdida y precisión para el modelo CNN-LSTM con atención.

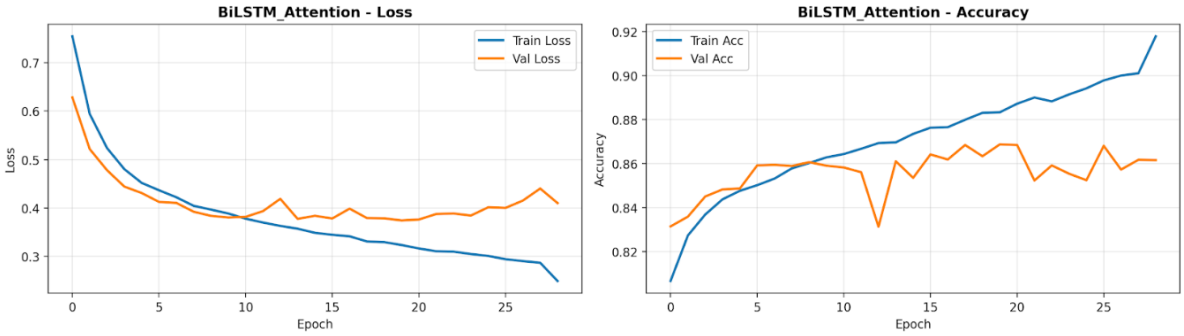


Figura 7.36: Curvas de pérdida y precisión para el modelo BiLSTM con atención.

La Figura 7.37 resume los valores finales de exactitud, precisión, recall y F1-score. Random Forest mantiene el mejor rendimiento global, con un F1-score de 0.833 y valores equilibrados entre precisión y recall. El Ensemble KNN obtiene un recall elevado de 0.902, junto con un F1-score de

0.614. Entre los modelos profundos, el CNN-LSTM con atención alcanza el mejor F1-score con un valor de 0.623, mientras que el BiLSTM obtiene 0.594. Estos resultados muestran que, aunque los modelos profundos no superan a los modelos clásicos balanceados, sí presentan una mejora respecto al experimento con Focal Loss.

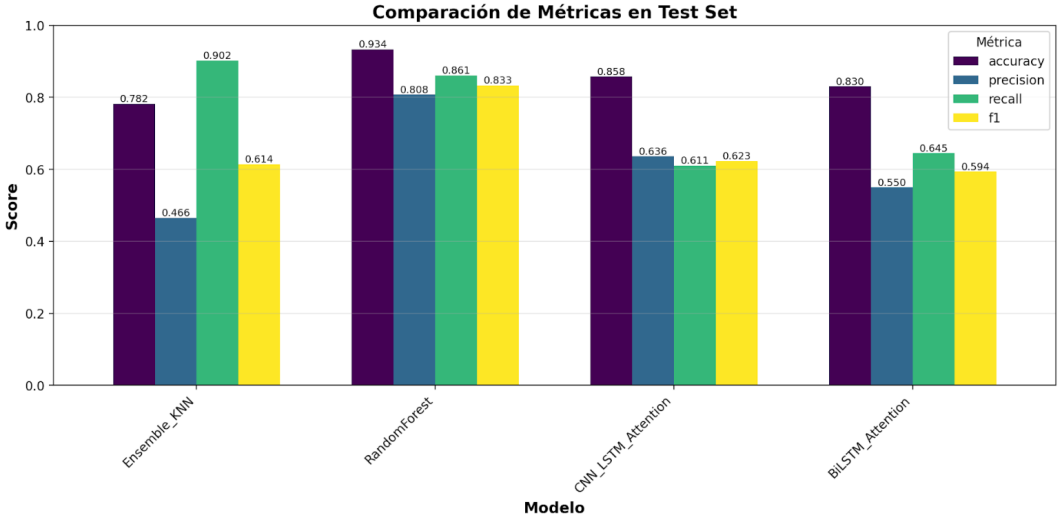


Figura 7.37: Comparación de métricas en el conjunto de prueba para el experimento con ADASYN y data augmentation.

7.2.5. Experimento 5 — ADASYN para ML y Focal Loss para DL

En este último experimento se mantuvo ADASYN como técnica de balanceo para los modelos de Machine Learning, mientras que los modelos de Deep Learning fueron entrenados empleando exclusivamente la función de pérdida Focal Loss con parámetros gamma = 2.5 y alpha = 0.75. A diferencia de los experimentos anteriores, en este caso no se utilizó ninguna técnica de data augmentation en los modelos profundos, con el objetivo de evaluar el efecto aislado de la pérdida enfocada en la clase minoritaria.

La Figura 7.38 muestra la distribución obtenida después de aplicar ADASYN. La clase Risk se expande nuevamente desde 5,861 hasta 23,983 muestras, alcanzando una distribución balanceada con respecto a la clase Safe.

Las curvas ROC para el conjunto de prueba se presentan en la Figura 7.39. El mejor AUC vuelve a corresponder al modelo Random Forest con un valor de 0.906, seguido por el Ensemble KNN con un AUC de 0.828. En los modelos de Deep Learning, el CNN-LSTM con atención obtiene un AUC de 0.678, mientras que el BiLSTM con atención alcanza un valor ligeramente superior de 0.702. En

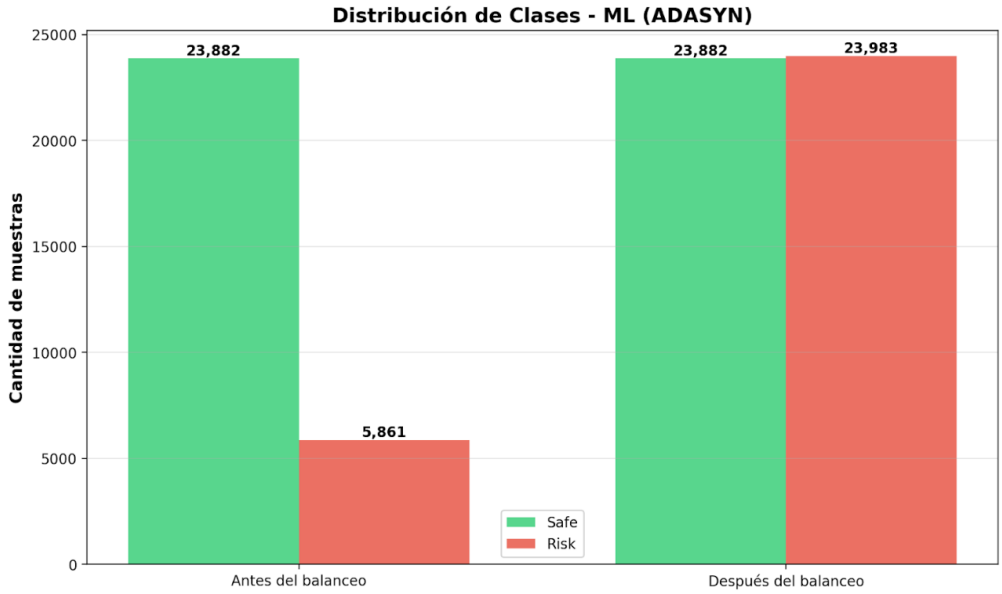


Figura 7.38: Distribución de clases antes y después del balanceo con ADASYN para los modelos de Machine Learning.

comparación con los experimentos 3 y 4, los valores de AUC disminuyen, lo que indica que el uso exclusivo de Focal Loss no es suficiente para sostener la capacidad discriminativa sin la presencia del data augmentation.

Las matrices de confusión de los modelos de Machine Learning se presentan en la Figura 7.40. El Ensemble KNN mantiene un 90.25 % de acierto en la clase Risk, mientras que Random Forest obtiene un 86.08 %. En la clase Safe, los valores corresponden a 75.32 % y 95.12 % respectivamente.

En los modelos profundos (Figura 7.41), el CNN-LSTM con atención clasifica correctamente el 43.11 % de las muestras Risk, mientras que el BiLSTM con atención alcanza el 46.01 %. Estos valores representan una reducción en la sensibilidad respecto a los experimentos donde se aplicó data augmentation.

Las curvas de entrenamiento se ilustran en las Figuras 7.42 y 7.43. El CNN-LSTM muestra mayor variabilidad entre las curvas de entrenamiento y validación, con oscilaciones en la pérdida y la precisión. El BiLSTM presenta un comportamiento algo más estable, aunque persisten oscilaciones moderadas.

La Figura 7.44 resume los valores finales de las métricas obtenidas. Random Forest continúa siendo el modelo con mejor desempeño global, alcanzando un F1-score de 0.833 y métricas equilibradas entre precisión y recall. El Ensemble KNN mantiene un recall muy elevado de 0.902 y un

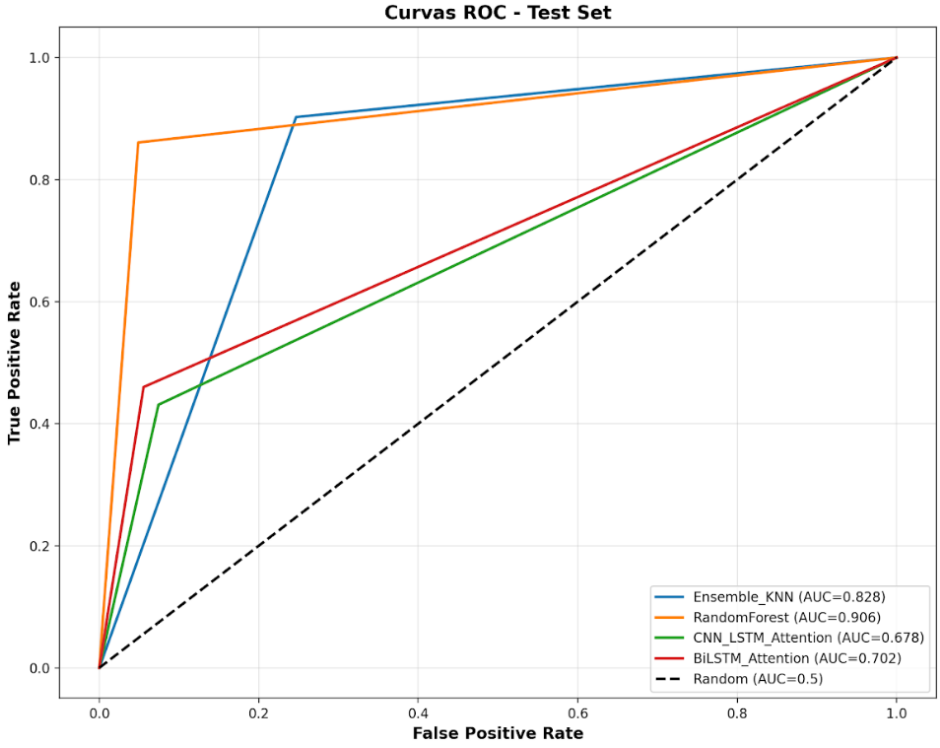


Figura 7.39: Curvas ROC de los modelos en el conjunto de prueba para el experimento con ADASYN y Focal Loss.

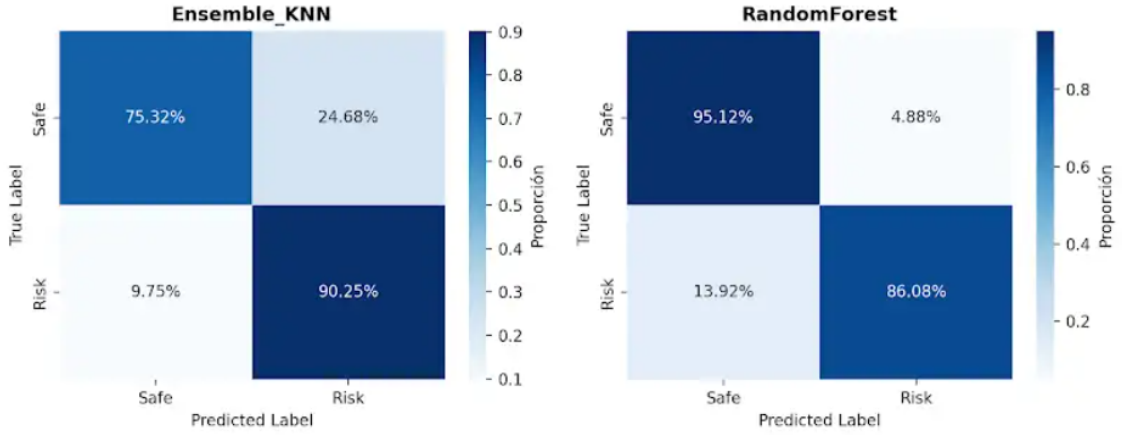


Figura 7.40: Matrices de confusión normalizadas para Ensemble KNN y Random Forest después de aplicar ADASYN.

F1-score de 0.614. En los modelos profundos, el BiLSTM obtiene el mejor F1-score con un valor de 0.543, ligeramente superior al CNN-LSTM, que alcanza un valor de 0.495. Estos resultados confir-

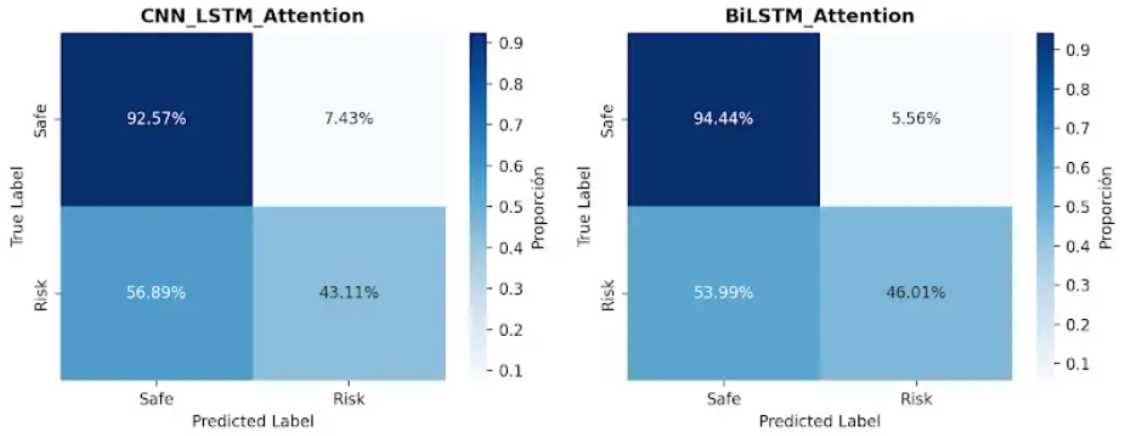


Figura 7.41: Matrices de confusión normalizadas para CNN-LSTM con atención y BiLSTM con atención.

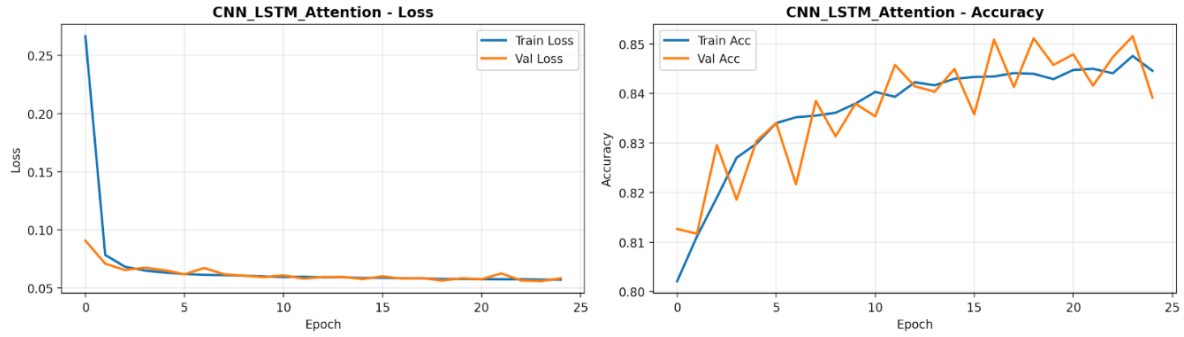


Figura 7.42: Curvas de pérdida y precisión para el modelo CNN-LSTM con atención.

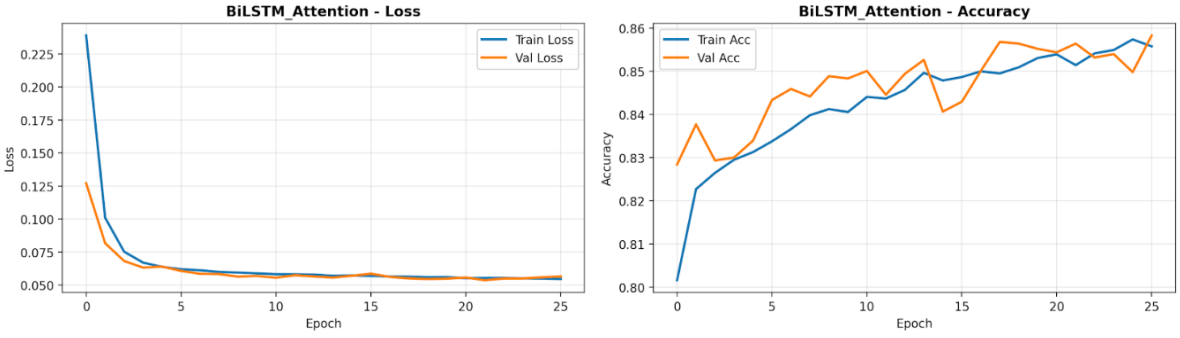


Figura 7.43: Curvas de pérdida y precisión para el modelo BiLSTM con atención.

man que la ausencia de data augmentation afecta negativamente la capacidad de generalización en ambos modelos profundos.

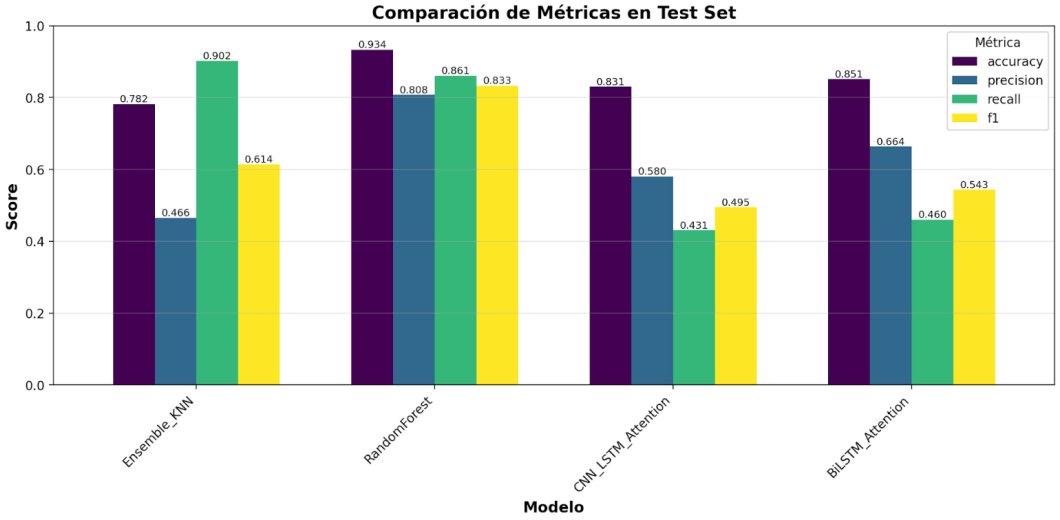


Figura 7.44: Comparación de métricas en el conjunto de prueba para el experimento con ADASYN y Focal Loss sin data augmentation.

7.3. Despliegue de la interfaz

El segundo módulo corresponde al backend, implementado en Python mediante el framework Flask. Su función principal es orquestar la ejecución de los modelos y manejar las peticiones provenientes del frontend a través de una API REST, ver la figura 7.45.

El servidor aloja los modelos y escaler, entrenados en el Colab, dentro del directorio models/, y expone rutas de comunicación. El diagrama 7.46 representa como el usuario interactúa con múltiples funcionalidades organizadas en un flujo de la app. El proceso comienza cuando el usuario carga señales EMG al sistema, ya sea mediante la función general de carga o específicamente subiendo archivos en formato .mat. Una vez cargadas, el usuario puede seleccionar las señales que desea procesar, lo cual extiende la funcionalidad de carga.

El núcleo del sistema es el caso de uso Çlasificar Señales", que requiere obligatoriamente que se hayan seleccionado las señales a procesar e incluye dos componentes fundamentales: el procesamiento de las señales EMG y el cálculo de métricas de rendimiento, para luego permitir la visualización de resultados a traves de gráficos y reportes de las clasificaciones.

Además de permitir la ejecución individual de cada modelo, el sistema incorpora un modo de clasificación dual que opera como un esquema de “doble opinión” automatizada. Esta funcionalidad permite combinar dos clasificadores especializados: uno optimizado para maximizar la detección de

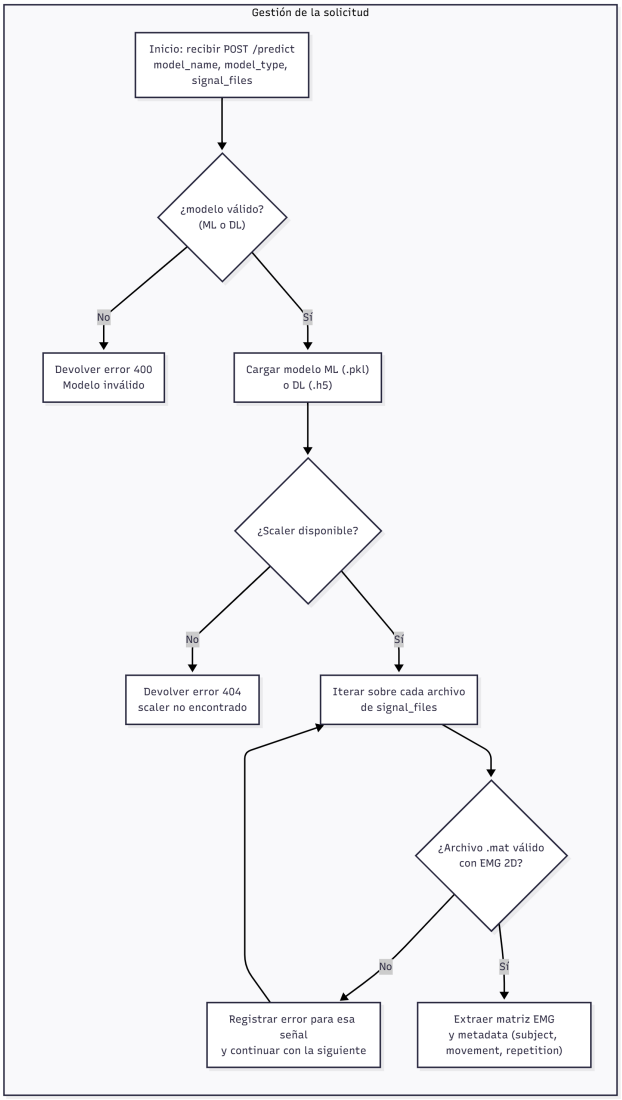


Figura 7.45: Estructura del backend, gestión de las solicitudes

casos de riesgo (mayor recall) y otro orientado a minimizar falsos positivos (precision). En este modo, el backend ejecuta primero el modelo especialista en identificar señales seguras; si este clasificador considera la señal como segura, se activa un segundo modelo orientado a recuperar posibles casos de riesgo que el primero podría pasar por alto. Esta arquitectura en cascada incrementa la robustez del sistema frente a señales ambiguas y permite ofrecer al usuario información más completa sobre cómo se tomó la decisión.

El pipeline de inferencia reproduce exactamente el preprocesamiento realizado en Colab para asegurar la consistencia entre entrenamiento y predicción. Los resultados de clasificación se devuel-

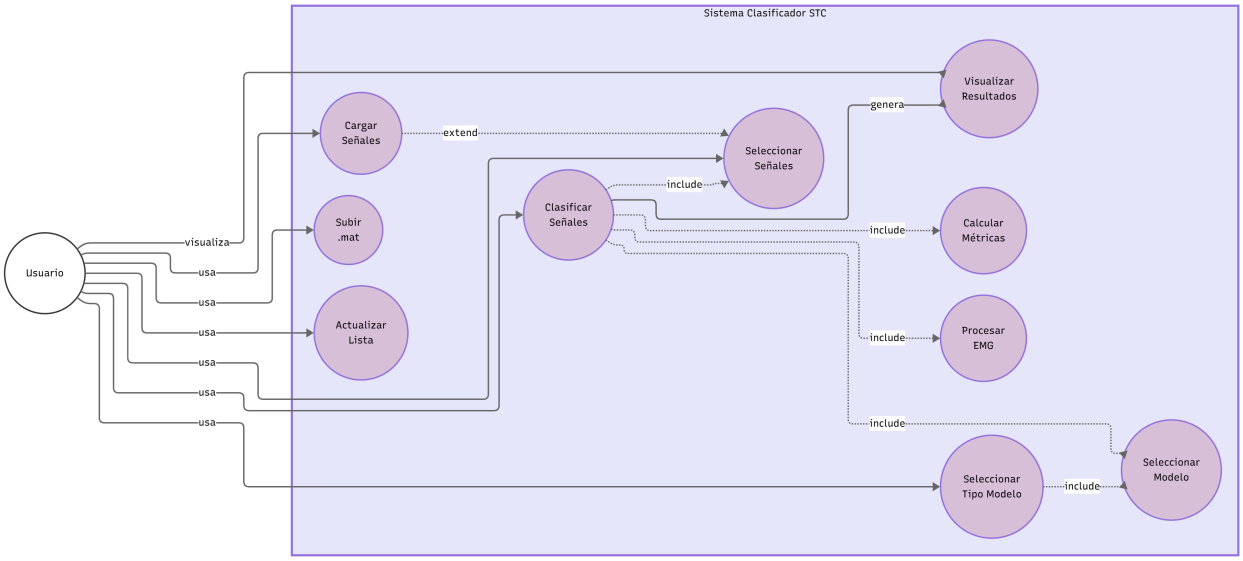


Figura 7.46: Diagrama de casos de uso

ven al frontend en formato JSON, incluyendo la etiqueta de clase, la probabilidad y los metadatos de la señal, ver la figura 7.47.

El tercer módulo corresponde a la interfaz de usuario, desarrollada mediante tecnologías web estándar (HTML, CSS y JavaScript). El archivo index.html contiene la estructura de la interfaz, que permite al usuario seleccionar un modelo y cargar una señal de prueba o una nueva señal (Figura 7.48). El script app.js gestiona la comunicación asíncrona con el backend mediante peticiones HTTP (fetch) hacia los endpoints del servidor Flask, como se aprecia en la figura 7.49.

Una vez recibida la respuesta, los resultados son procesados y visualizados dinámicamente en la página mediante gráficos interactivos en Canvas Charts. La interfaz muestra tanto la probabilidad de clasificación como el tipo de movimiento identificado (seguro o de riesgo), como se observa en el ejemplo de las Figuras 7.50 y 7.51. En este ejemplo se seleccionó la clasificacion dual, que usa los modelos Random Forest y Ensemble_KNN del ejercicio 4 (los que obtuvieron mejores resultados clasificando Safe y Risk respectivamente), luego se cargó la repetición 5 de los movimientos 1, 13 y 14 del sujeto 20. Se puede observar como el primer movimiento es clasificado como "seguroçomo consenso entre los modelos, mientras que los movimientos 13 y 14 fueron clasificados como riesgoçon una certeza de 80.9 % y 76.9 % respectivamente de acuerdo con el Random Forest.

Esta arquitectura modular facilita la trazabilidad del proceso completo, desde el entrenamiento hasta la inferencia, garantizando la reproducibilidad del pipeline y la independencia de cada componente.

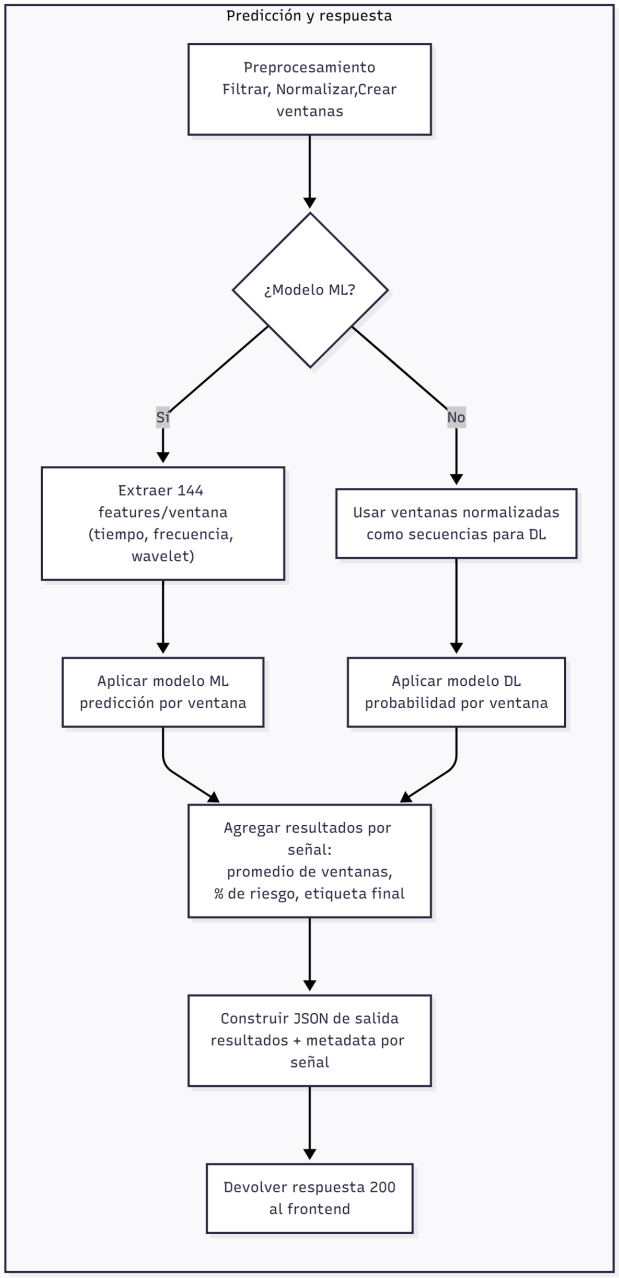


Figura 7.47: Estructura del backend, predicción y respuesta

7.4. Discusión

Los resultados obtenidos a lo largo de los cinco experimentos permiten observar cómo influye el desbalance de clases y la técnica de balanceo en el desempeño de los modelos para la detección de

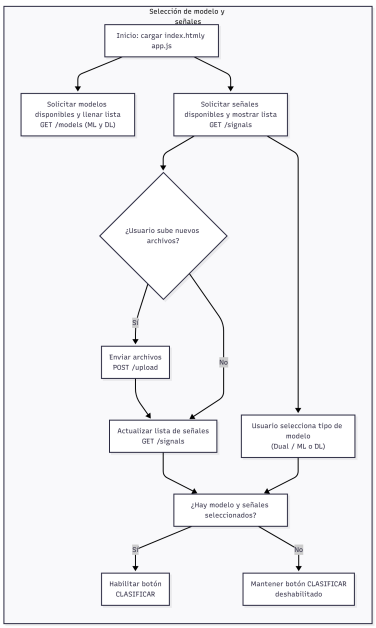


Figura 7.48: Estructura de la interfaz web, selección del modelo y las señales

patrones. Como se dijo anteriormente, la relación inicial entre clases (23 882 muestras Safe frente a 5 861 Risk) genera un escenario donde la precisión global tiende a ser alta incluso cuando los modelos fallan en identificar la clase minoritaria. Por esto, el análisis se centró en métricas sensibles al desbalance, como el recall y el F1-score.

En el experimento sin balanceo, los modelos de Machine Learning y de Deep Learning muestran comportamientos distintos. Random Forest obtiene la mayor especificidad pero una sensibilidad reducida, clasificando correctamente el 99 % de las muestras Safe, pero apenas el 57.8 % de las muestras Risk. Esta tendencia se mantiene en el Ensemble KNN, que presenta un recall cercano al 48 %. En contraste, las arquitecturas profundas logran valores de recall ligeramente superiores, como en el caso de la CNN-LSTM, que alcanza aproximadamente 63 % de detección para la clase de riesgo. No obstante, esto se obtiene con una precisión considerablemente menor en comparación con los modelos clásicos. Este comportamiento evidencia que los modelos basados en características manuales tienden a reproducir el sesgo del conjunto de entrenamiento, mientras que las redes profundas, al operar directamente sobre secuencias crudas, capturan algunas variaciones de la señal asociadas al riesgo, aunque también aumenta los falsos positivos. Este resultado muestra que un sistema sin balanceo no es viable, ya que presentaría un número excesivo de falsos negativos.

El uso de SMOTE en el experimento 2 produce una mejora significativa para los modelos clásicos. El Random Forest aumenta su F1-score hasta aproximadamente 0.83 y alcanza valores de recall cercanos al 84 %, demostrando una mayor capacidad de identificar eventos de riesgo. El Ensemble

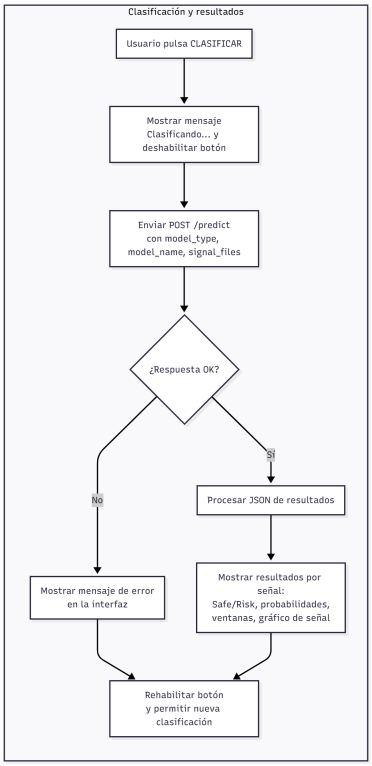


Figura 7.49: Estructura de la interfaz web, clasificación y resultados

KNN también mejora, alcanzando un recall superior al 86 %. Por el contrario, los modelos de Deep Learning no reflejan este beneficio, tanto la CNN-LSTM como el BiLSTM reducen su F1-score respecto a la línea base. Esta diferencia puede explicarse por el hecho de que SMOTE opera en el espacio estático de características extraídas, lo que favorece unicamente a los métodos clásicos.

Cuando se incorpora ADASYN junto con Focal Loss y data augmentation en el experimento 3, ADASYN mejora nuevamente el rendimiento de los modelos clásicos, con F1-scores de 0.83 para Random Forest y 0.61 para Ensemble KNN. Sin embargo, en los modelos profundos se observa una leve disminución en la sensibilidad al riesgo, con valores entre 41 % y 47 %, inferiores a los alcanzados sin balanceo. Aunque Focal Loss penaliza más fuertemente los errores en la clase minoritaria, este ajuste no es suficiente si las muestras continúan siendo poco representativas de la variabilidad temporal real de la señal.

El experimento 4 muestra un punto intermedio relevante, ADASYN para Machine Learning y únicamente data augmentation para Deep Learning. Los modelos profundos experimentan una mejora notable en comparación con el experimento anterior. La CNN-LSTM alcanza un F1-score de 0.62 y el BiLSTM llega a 0.59, con sensibilidades de 61 % y 65 %, respectivamente. La ausencia de Focal Loss parece estabilizar el aprendizaje, permitiendo que el data augmentation aporte variabili-

Figura 7.50: Fronted de la App

dad fisiológicamente plausible sin producir oscilaciones en la función de pérdida. Esto muestra que en el caso de redes profundas, la regularización basada en transformaciones temporales puede ser más efectiva que la manipulación de la función de pérdida.

Por otro lado, en el experimento 5 se evaluó ADASYN para los modelos clásicos y Focal Loss para los modelos profundos. Tal como se anticipaba a partir de los experimentos previos, los modelos de Machine Learning mantienen un rendimiento robusto, con Random Forest nuevamente en torno a F1 = 0.83 y con alta sensibilidad. Sin embargo, el desempeño de los modelos profundos tiende a degradarse, alcanzando F1-scores de 0.49 para CNN-LSTM y 0.54 para BiLSTM. Las matrices de confusión confirman una reducción en la detección de la clase Risk, con valores alrededor del 43 %

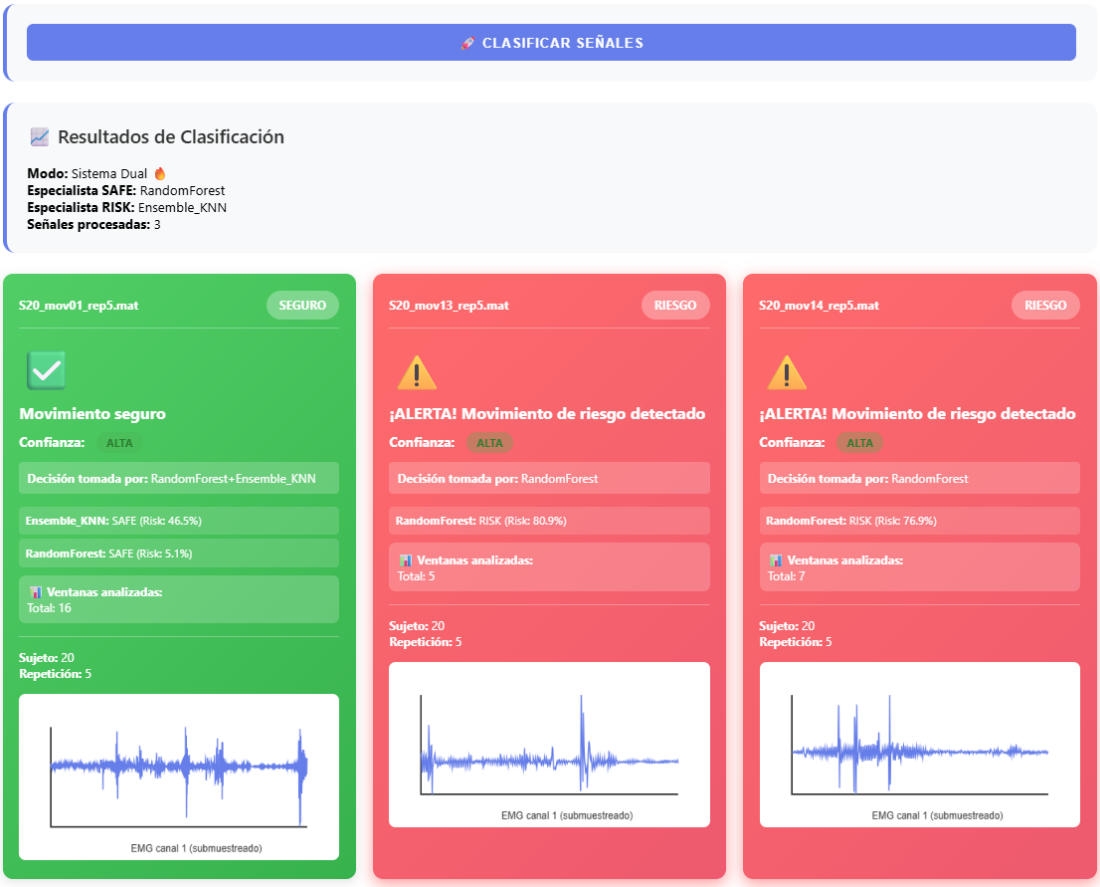


Figura 7.51: Ejemplo de uso de la App

y 46 %. Esto evidencia que el uso aislado de Focal Loss sin data augmentation no permite que la red generalice adecuadamente.

Es importante resaltar que estos comportamientos están alineados con el análisis espectral realizado. En efecto las diferencias globales entre las clases SAFE y RISK en términos de potencia, frecuencia pico y frecuencia media fueron pequeñas y estadísticamente no significativas, lo que indica que ambas clases comparten patrones espectrales similares. Esta falta de separabilidad directa explica por qué los modelos basados únicamente en características manuales presentan dificultades para identificar la clase minoritaria sin técnicas de balanceo, y también por qué las redes profundas requieren data augmentation para capturar variaciones temporales más sutiles que no están presentes de forma evidente en el espectro promedio.

En cuanto a las limitaciones, la estrategia utilizada para segmentar los datos por repeticiones garantiza independencia temporal entre las ventanas, pero no independencia entre sujetos. Los mis-

mos participantes aparecen en los conjuntos de entrenamiento, validación y prueba, por lo que los modelos pueden estar aprendiendo patrones específicos de cada sujeto y no solo características relacionadas con el riesgo del movimiento. Adicionalmente, la base de datos NinaPro DB2 fue diseñada para la clasificación multiclase de gestos de la mano en sujetos sanos, y no para tareas relacionadas al síndrome del túnel carpiano ni para una tarea binaria SAFE/RISK.

Finalmente, la carga computacional asociada al procesamiento de señales EMG de alta resolución y al entrenamiento de modelos profundos restringió la exploración de arquitecturas más complejas y la aplicación sistemática de búsquedas extensivas de hiperparámetros o esquemas de validación cruzada por sujeto. Futuros trabajos deberían considerar particiones exclusivas por participante, la inclusión de bases de datos especificas para tareas relacionadas con STC y una exploración más amplia de arquitecturas y estrategias de regularización que permitan evaluar con mayor profundidad la capacidad de generalización de los modelos.

Capítulo 8

Conclusiones

8.1. Conclusiones

En este trabajo se cumplió satisfactoriamente el objetivo general de desarrollar una herramienta para la identificación de movimientos de la mano relacionados con el síndrome del túnel carpiano mediante procesamiento de señales de electromiografía y aprendizaje automático. A través del desarrollo sistemático de cada objetivo específico, se construyó un sistema completo de clasificación de riesgo basado en señales sEMG que permite detectar movimientos asociados al desarrollo de STC con alta sensibilidad.

El primer objetivo específico se alcanzó mediante la selección y gestión de la base de datos NinaPro DB2 (Ejercicio B) utilizando el método AHP, considerando número de sujetos, calidad de adquisición, frecuencia de muestreo (2000 Hz) y documentación disponible. Se implementó un pipeline reproducible de filtrado (20–450 Hz, notch a 50 Hz), segmentación en ventanas de 500 ms con 25 % de solapamiento y extracción de 144 características en dominios temporal, frecuencial y tiempo-frecuencia. A partir de los 40 sujetos se generaron 29743 ventanas para entrenamiento, 7429 para validación y 7354 para test, estableciendo una relación estable de 1:4 entre la clase Risk (movimientos 13–16) y la clase Safe (movimientos 1–12 y 17). El análisis espectral global reveló frecuencias pico de $69{,}61 \pm 44{,}19$ Hz (SAFE) y $74{,}26 \pm 46{,}88$ Hz (RISK), con potencia total de $1{,}59 \times 10^{-1}$ y $1{,}71 \times 10^{-1}$ respectivamente, aunque sin diferencias significativas, confirmando la necesidad de modelos de aprendizaje automático.

El segundo objetivo específico se cumplió mediante la implementación y entrenamiento de modelos de Machine Learning (Random Forest, Ensemble Subspace KNN) y Deep Learning (CNN–LSTM con atención, BiLSTM). Se evaluaron tres esquemas de balanceo: sin balanceo, con SMOTE y con ADASYN. Los modelos clásicos entrenados sobre las 144 características manuales y balanceados con ADASYN demostraron el mejor desempeño. Random Forest alcanzó un F1-score de 0.833 y un AUC de 0.906, mientras que Ensemble KNN logró una sensibilidad del 90.2 % para la clase Risk. Las arquitecturas profundas, aunque capaces de capturar patrones temporales de la señal cruda, obtuvieron resultados inferiores con la configuración actual (CNN–LSTM: F1 = 0.623, sensibilidad = 61.1 %; BiLSTM: F1 = 0.594, sensibilidad = 64.5 % con data augmentation).

El tercer objetivo específico se logró mediante el análisis comparativo de las métricas de entrenamiento, priorizando sensibilidad, F1-score y AUC sobre la exactitud global. Los experimentos sin balanceo evidenciaron el efecto del desbalance: Random Forest alcanzó 94.5 % de exactitud pero

solo 57.9 % de sensibilidad en Risk. Con ADASYN, Random Forest incrementó su sensibilidad hasta 86.1 % manteniendo un F1-score de 0.833, demostrando que las técnicas de sobremuestreo son esenciales para detectar movimientos de riesgo. El Ensemble KNN con ADASYN obtuvo la mayor sensibilidad (90.2 %), estableciendo que los modelos clásicos balanceados constituyen la alternativa más robusta para esta tarea, detectando más de 8 de cada 10 ventanas de riesgo.

El cuarto objetivo específico se cumplió con el desarrollo de un prototipo funcional en Flask que integra los modelos entrenados en un entorno web. La aplicación incorpora un modo de clasificación dual que combina Random Forest (mejor desempeño con la clase Safe) y Ensemble KNN (mejor desempeño con Risk) para obtener un dictamen más robusto. La interfaz permite cargar nuevas señales, reproducir el pipeline de preprocesamiento completo y visualizar probabilidades de riesgo, constituyendo una herramienta de apoyo potencial para estudios de ergonomía y futuras investigaciones sobre clasificación de movimientos asociados a STC.

Desde una perspectiva clínica y fisiopatológica, los movimientos clasificados como de riesgo han sido ampliamente asociados con incrementos de la presión intracarpiana, favoreciendo la compresión progresiva del nervio mediano. La capacidad del sistema propuesto para identificar estos gestos a partir de patrones de actividad muscular registrados mediante sEMG sugiere que las alteraciones biomecánicas que preceden al daño neurológico tienen firmas electromiográficas diferenciables. En este sentido, la herramienta desarrollada no se plantea como un método diagnóstico, sino como un sistema de apoyo orientado a la detección temprana de patrones motores potencialmente perjudiciales, con aplicaciones en prevención y vigilancia ergonómica.

Los resultados demuestran que el sistema desarrollado cumple de manera consistente los objetivos propuestos, ofreciendo un enfoque robusto para la identificación de movimientos asociados a un mayor riesgo biomecánico del síndrome del túnel carpiano. Los modelos clásicos balanceados mediante ADASYN alcanzaron F1-scores superiores a 0.83 y sensibilidades entre el 86 % y el 90 %, evidenciando una alta capacidad para detectar movimientos de riesgo en etapas tempranas. Si bien las arquitecturas profundas evaluadas presentaron un desempeño inferior bajo la configuración actual, sus resultados sugieren que podrían mejorar con bases de datos diseñadas específicamente para escenarios binarios SAFE/RISK.

De este modo, el proyecto sienta una base sólida para el desarrollo de herramientas no invasivas orientadas al monitoreo continuo, la prevención y el apoyo a la toma de decisiones en contextos clínicos y ocupacionales. En un escenario futuro, el sistema propuesto podría integrarse con dispositivos de hardware portátiles basados en electromiografía de superficie, permitiendo el seguimiento continuo de patrones de movimiento en entornos laborales reales. Este tipo de integración facilitaría la identificación de posturas biomecánicamente desfavorables en poblaciones con alta exposición al riesgo, como apoyo para generar intervenciones ergonómicas y programas de prevención, contribuyendo a reducir la incidencia del STC.

Capítulo 9

Trabajos futuros

Como proyección futura de este trabajo se identifican diversas áreas que permitirán fortalecer y ampliar el alcance de la herramienta de clasificación de movimientos asociados a STC basado en señales electromiográficas (sEMG). A pesar del desempeño alcanzado por los modelos, existen limitaciones intrínsecas tanto en el conjunto de datos como en el proceso de adquisición de señales, que restringen la generalización del modelo hacia contextos clínicos o laborales.

Primero, se reconoce que la base de datos Ninapro presenta ciertos inconvenientes para la clasificación binaria propuesta en este trabajo entre movimientos de riesgo (flexión, extensión, desviación radial y ulnar) y la clase safe. Entre ellos destacan el desbalance pronunciado entre ambas clases, el mal etiquetado de algunas señales, así como segmentos incompletos o inconsistentes en determinados movimientos. Estas condiciones pueden inducir sesgos en el entrenamiento y afectar la estabilidad de los modelos. En consecuencia, se plantea como trabajo futuro la exploración de otros métodos para manejar desequilibrios severos entre clases.

Además, dado que el campo de la electromiografía superficial carece de un protocolo estandarizado de adquisición reconocido globalmente, se propone contribuir a la creación de una base de datos especializada y balanceada orientada al estudio del Síndrome del Túnel Carpiano (STC). Dicha base debería incluir movimientos específicamente asociados a la aparición del STC, información demográfica y variables clínicas relevantes, permitiendo validar modelos en condiciones más cercanas al entorno real.

Ademas se considera que otra área de trabajo es la integración directa entre la aplicación desarrollada y hardware para captura electromiográfica, permitiendo la transmisión en tiempo real de señales EMG desde diferentes dispositivos comerciales o prototípados. Este avance demandará optimizar la comunicación entre capas de software y hardware, priorizando la seguridad del entorno de transmisión, la latencia mínima en el envío de datos y una experiencia de usuario fluida e intuitiva. Tal conexión abriría la posibilidad de implementar el sistema como una herramienta práctica para el monitoreo continuo.

Desde otra perpectiva se visualiza la incorporación de nuevas variables fisiológicas y biomecánicas al proceso de clasificación. Esto incluye ángulos articulares, mediciones de fuerza, vibraciones, y parámetros cinemáticos que, combinados con la información de EMG, podrían mejorar la sensibilidad del sistema ante alteraciones motoras tempranas vinculadas al STC. La fusión de estas modalidades permitiría transitar hacia modelos híbridos que integren sensores de movimiento (IMU) y de presión,

ampliando la capacidad diagnóstica del sistema.

Finalmente, se espera avanzar hacia la validación clínica del sistema en entornos reales. Este paso permitirá evaluar no solo la precisión técnica, sino también la utilidad práctica de la herramienta en contextos de salud ocupacional, orientando su diseño hacia una interfaz más intuitiva y accesible. En conjunto se pretende consolidar un ecosistema integral de análisis electromiográfico aplicado a la prevención de trastornos músculo esqueléticos laborales, como lo es el Síndrome del Túnel Carpiano.

Capítulo 10

Anexos

Anexo 1 – Manual de Usuario

HeClaMoSTC

Herramienta de Clasificación de Movimientos para Síndrome del Túnel Carpiano

Manual de Usuario para Descarga e Instalación

Autora:
Karen Nicolle Arango Valencia

Universidad:
Pontificia Universidad Javeriana - Cali

Fecha: Noviembre 2025

Índice

1. Introducción

1.1. ¿Qué hace HeClaMoSTC?

HeClaMoSTC (Herramienta de Clasificación de Movimientos para Síndrome del Túnel Carpiano) es un sistema inteligente de clasificación automática diseñado para detectar movimientos de riesgo asociados al Síndrome del Túnel Carpiano (STC) mediante el análisis de señales electromiográficas (EMG) de superficie.

El sistema analiza señales EMG de 12 canales y clasifica automáticamente cada movimiento en dos categorías: **SEGURO** y **RIESGO**, en función de patrones aprendidos a partir de la base de datos NinaPro DB2 (Ejercicio B).

Clasificación de movimientos

- **Movimientos de RIESGO** (asociados a sobrecarga biomecánica en el túnel carpiano):
  - Movimiento 13: Flexión de muñeca.
  - Movimiento 14: Extensión de muñeca.
  - Movimiento 15: Desviación radial.
  - Movimiento 16: Desviación ulnar.
- **Movimientos SEGUROS**:
  - Movimientos 1–12: Diversos agarres y gestos de mano.
  - Movimiento 17: Extensión de muñeca con mano cerrada (considerado seguro en este contexto).

1.2. Características principales

HeClaMoSTC ofrece las siguientes características:

- **Interfaz web intuitiva**: no requiere conocimientos de programación; toda la interacción se realiza a través de un navegador web.
- **Múltiples modelos**:
  - Machine Learning (ML): Ensemble KNN y Random Forest.
  - Deep Learning (DL): CNN + LSTM + Attention y BiLSTM + Attention.

- **Sistema Dual**: combina dos modelos especializados para mejorar la detección:
  - *Especialista SAFE*: optimizado para detectar movimientos seguros.
  - *Especialista RISK*: optimizado para detectar movimientos de riesgo.
- **Pipeline automático**:
  - Filtrado (Butterworth 20–450 Hz + notch 50 Hz).
  - Normalización Z-score.
  - Segmentación en ventanas.
  - Extracción de características (para ML).
  - Clasificación y agregación de resultados.
- **Visualización**:
  - Gráficas de señales EMG.
  - Estadísticas por ventana.
  - Probabilidades de clasificación.
  - Niveles de confianza y detalle del Sistema Dual.
- **Compatibilidad con archivos propios**: permite cargar archivos `.mat` con señales EMG personalizadas.

2. Requisitos del sistema

2.1. Requisitos de hardware

Mínimo:

- Procesador dual-core a 2.0 GHz o superior.
- 4 GB de RAM.
- 1 GB de espacio libre en disco..

Para entrenamiento de modelos:

- GPU NVIDIA con soporte CUDA (opcional, se usa Google Colab).
- 16 GB de RAM.
- 10 GB de espacio libre en Google Drive.

2.2. Requisitos de software

Sistema operativo: El sistema fue desarrollado en Windows 11. En caso de usar macOS o Linux se deben revisar la compatibilidad de paquetes y comandos.

Python:

- Versión 3.8, 3.9, 3.10 o 3.11.
- `pip` instalado.

Conexión a Internet:

- Necesaria para instalación de dependencias.
- Necesaria para entrenamiento en Google Colab.
- Opcional para el uso local de la aplicación (una vez instalada).

2.3. Dependencias de Python

Las siguientes librerías se instalan desde `requirements.txt`:

Cuadro 1: Dependencias principales de HeClaMoSTC

Librería	**Versión**	**Propósito**
Flask	3.0.0	Servidor web backend
Flask-CORS	4.0.0	Manejo de CORS
TensorFlow	2.15.0	Modelos de Deep Learning
Keras	2.15.0	API de alto nivel para DL
scikit-learn	1.3.2	Modelos de Machine Learning
NumPy	1.24.3	Cálculo numérico
SciPy	1.11.4	Procesamiento de señales
pandas	2.1.4	Manejo de datos
imbalanced-learn	0.11.0	Técnicas de balanceo de clases
PyWavelets	1.5.0	Análisis wavelet
matplotlib	3.8.2	Gráficas (notebooks)
seaborn	0.13.0	Visualización avanzada

3. Descarga e instalación

3.1. Descargar desde GitHub

Paso 1: acceder al repositorio

Abre el navegador y visita:

https://github.com/proyectoSTC/HeClaMoSTC

Paso 2: descargar el código

Opción A: descarga directa (usuarios sin Git)

a) Haz clic en el botón verde `Code`.

b) Selecciona `Download ZIP`.

c) Guarda el archivo e inmediatamente después descomprímelo.

d) Mueve la carpeta a una ruta conocida, por ejemplo:

- Windows: `C:\Users\TuUsuario\Documentos\HeClaMoSTC`
- macOS: `/Users/TuUsuario/Documentos/HeClaMoSTC`
- Linux: `/home/TuUsuario/Documentos/HeClaMoSTC`

Opción B: usando Git (usuarios avanzados)

```
git clone https://github.com/proyectoSTC/HeClaMoSTC.git
cd HeClaMoSTC
```

3.2. Instalación en Windows

Paso 1: verificar Python

Abre `CMD` y ejecuta:

```
python --version
```

Deberías ver algo como `Python 3.10.x`. Si no está instalado:

- Descarga desde https://www.python.org/downloads/.
- Marca la casilla `Add Python to PATH`.
- Completa la instalación y reinicia CMD.

Paso 2: ir a la carpeta del proyecto

```
cd C:\Users\TuUsuario\Documentos\HeClaMoSTC
```

Paso 3: crear entorno virtual (recomendado)

```
python -m venv venv
venv\Scripts\activate
```

Paso 4: instalar dependencias

```
pip install -r requirements.txt
```

Si hay errores, actualiza `pip` y repite:

```
python -m pip install --upgrade pip
pip install -r requirements.txt
```

Paso 5: verificación rápida

```
python -c "import flask, tensorflow, sklearn; print('Instalación exitosa')"
```

3.3. Estructura de carpetas

Tras la instalación, la estructura recomendada es:

```
HeClaMoSTC/

 frontend/              (Interfaz web)
    index.html
    app.js

 models/                (Modelos entrenados)
    ensemble_knn.pkl
```

```
    random_forest.pkl
    cnn_lstm_attention.keras
    bilstm_attention.keras
    scaler.pkl
    metadata.json

signals/                (Señales de prueba)
    *.mat

notebooks/              (Notebooks de Colab/Jupyter)
    HeClaMoSTC.ipynb
    EMG_Spectral_Analysis.ipynb

venv/                   (Entorno virtual)
server.py               (Servidor backend Flask)
requirements.txt        (Dependencias)
README.md               (Documentación)
```

Importante:

- Copiar `index.html` y `app.js` a la carpeta `frontend/`.
- Copiar todos los modelos (`.pkl`, `.keras`) junto con `scaler.pkl` y `metadata.json` a `models/`.
- Copiar señales de prueba a `signals/`. Para descargar las señales utilizadas en el proyecto hacerlo a traves del link: https://drive.google.com/drive/folders/1VizpksUwSnVtW46p8OzCbKDbdEurR-s?usp=sharing. (Contiene la carpeta Train, para usar el Colab y la carpeta Test para probar la App)

Sin `scaler.pkl` ni `metadata.json` el sistema no podrá funcionar correctamente.

4. Uso de la aplicación web

4.1. Iniciar el servidor

Paso 1: activar entorno virtual

Windows:

```
venv\Scripts\activate
```

Paso 2: ejecutar el servidor

```
python server.py
```

Deberías ver algo similar a:

```
======================================================================
SERVIDOR CLASIFICADOR STC
======================================================================

 Rutas: C:\Users\Karen\HeClaMoSTC\models
 Window: 500ms, Overlap: 25.0%

 Sistema Dual DISPONIBLE:
   SAFE: RandomForest (precision 0.952)
   RISK: Ensemble_KNN (recall 0.897)

 http://localhost:5000
======================================================================

 * Serving Flask app 'server'
 * Debug mode: on
 * Running on http://0.0.0.0:5000
```

Paso 3: abrir el navegador

Visita:

```
http://localhost:5000
```

Notas:

- Mantener la terminal abierta mientras se usa la aplicación.
- Para detener el servidor: `Ctrl + C`.
- Si el puerto 5000 está ocupado, se puede cambiar en `server.py`.

4.2. Interfaz principal

La interfaz se organiza en varias secciones:

- **Encabezado**: título y breve descripción del sistema.
- **Modo de clasificación**:
  - **Modelo independiente**: permite seleccionar un único modelo (ML o DL).
  - **Sistema Dual**: combina automáticamente dos modelos ML especialistas.
- **Selección de modelo**:
  - Tipo de modelo: ML o DL.
  - Modelo específico: `ensemble_knn`, `random_forest`, `cnn_lstm_attention` o `bilstm_attentio`
- **Señales de prueba**:
  - Lista de archivos `.mat` disponibles.
  - Botón para cargar archivos propios.
  - Botón para actualizar la lista.
- **Botón de clasificación**:
  - Se habilita tras elegir modelo y señales.
- **Área de resultados**:
  - Tarjetas de resultados por señal.
  - Gráficas y estadísticas.

4.3. Clasificación paso a paso

Modo independiente

1. Elegir **Modo de Clasificación**: "Modelo Independiente".
2. Elegir **tipo de modelo**: ML o DL.
3. Seleccionar el modelo concreto:
   - ML: `ensemble_knn` o `random_forest`.
   - DL: `cnn_lstm_attention` o `bilstm_attention`.
4. Seleccionar una o varias señales en la lista.
5. Pulsar **"CLASIFICAR SEÑALES"**.
6. Esperar a que se muestren las tarjetas de resultados.

Sistema Dual (recomendado)

1. Elegir **Modo de Clasificación**: "Sistema Dual".
2. Verificar la información del sistema:
   - Modelo especialista SAFE.
   - Modelo especialista RISK.
   - Fecha de los metadatos (`metadata.json`).
3. Seleccionar las señales a clasificar.
4. Pulsar **"CLASIFICAR SEÑALES"**.
5. Revisar:
   - Clasificación final.
   - Nivel de confianza (ALTA / MEDIA).
   - Contribución de cada modelo.

4.4. Carga de señales propias

1. Preparar los archivos `.mat` con la estructura adecuada (ver siguiente sección).
2. En la interfaz, hacer clic en **"Cargar .mat desde tu PC"**.
3. Seleccionar uno o varios archivos.
4. Confirmar la subida.
5. Verificar que aparecen en la lista de señales.
6. Clasificar normalmente.

Los archivos subidos se almacenan en la carpeta `signals/`. Se pueden eliminar manualmente desde el sistema de archivos y luego actualizar la lista desde la interfaz.

4.5. Formato de archivos `.mat`

Los archivos `.mat` deben incluir:

- `emg` (obligatoria):
  - Matriz de tamaño $[n\_muestras \times 12]$.
  - Tipo numérico (`float32`, `float64`, `int16`, etc.).
  - Cada columna es un canal EMG.
- `subject` (opcional): ID del sujeto.
- `stimulus` / `restimulus` (opcional):
  - Etiqueta de movimiento (1–17).
  - 13–16: movimientos RISK.
  - Otros: movimientos SAFE.
- `repetition` / `rerepetition` (opcional): número de repetición (1–6).

Requisitos:

- Frecuencia de muestreo: 2000 Hz.
- 12 canales EMG.

- Duración mínima recomendada: 1 segundo (2000 muestras).

Ejemplo en MATLAB:

```
emg = randn(10000, 12);
subject = 1;
stimulus = 13;
repetition = 1;
save('mi_senal.mat', 'emg', 'subject', 'stimulus', 'repetition');
```

Ejemplo en Python:

```
import numpy as np
from scipy.io import savemat

emg = np.random.randn(10000, 12).astype(np.float32)
subject = 1
stimulus = 13
repetition = 1

savemat('mi_senal.mat', {
    'emg': emg,
    'subject': subject,
    'stimulus': stimulus,
    'repetition': repetition
})
```

5. Entrenamiento de modelos en Google Colab

5.1. Acceder al notebook

1. Abrir https://colab.research.google.com/.
2. Iniciar sesión con una cuenta de Google.
3. Cargar el notebook:
   - Desde GitHub: usar la pestaña GitHub e introducir la URL del repositorio `HeClaMoSTC`.
   - O subir el archivo `HeClaMoSTC.ipynb`.

5.2. Configurar GPU y montar Google Drive

1. En Colab: `Entorno de ejecución` → `Cambiar tipo de entorno de ejecución`.
2. Seleccionar "GPU" en *Acelerador de hardware*.
3. Guardar cambios.

Verificar GPU:

```
!nvidia-smi
```

Montar Google Drive:

```
from google.colab import drive
drive.mount('/content/drive')
```

5.3. Preparar datos en Google Drive

Estructura recomendada:

```
Mi unidad/
 DB2_E1_only/
     train/
        S1_E1_A1.mat
        ...
```

```
    test/
        S1_E1_A1.mat
        ...
```

Cada archivo `.mat` debe contener al menos:

- `emg`: $[n\_muestras \times 12]$.
- `restimulus`: vector con el movimiento.
- `rerepetition`: vector con la repetición.

Verificación rápida:

```
from pathlib import Path

data_dir = Path('/content/drive/MyDrive/DB2_E1_only/train')
mat_files = list(data_dir.glob('*.mat'))

print(f"Archivos encontrados: {len(mat_files)}")
for f in mat_files[:5]:
    print(" -", f.name)
```

5.4. Configurar el notebook

Rutas principales

```
class Config:
    BASE_DIR = Path('/content/drive/MyDrive')
    DATA_DIR = BASE_DIR / 'DB2_E1_only'
    TRAIN_DIR = DATA_DIR / 'train'
    SAVE_DIR = BASE_DIR / 'New_ML_DL_models_stc_optimized'
```

Selección de sujetos y modelos

```
USE_ALL_SUBJECTS = True
SELECTED_SUBJECTS = [1,2,3,4,5,6,7,8,9,10,
                     11,12,13,14,15,16,17,18,19,20]

SELECTED_MODELS = ['1', '2', '3', '4']
```

```
# '1': Ensemble KNN
# '2': Random Forest
# '3': CNN+LSTM+Attention
# '4': BiLSTM+Attention
```

Técnicas de balanceo

```
ML_BALANCE_TECHNIQUE = 'adasyn'
DL_BALANCE_TECHNIQUE = 'augment_only'
```

5.5. Ejecutar el entrenamiento

Se puede usar `Entorno de ejecución` → `Ejecutar todas` o ir celda por celda.

El notebook:

- Carga y preprocesa los datos.
- Entrena los modelos ML y DL.
- Genera métricas (accuracy, precision, recall, F1, AUC).
- Guarda los modelos y artefactos.

5.6. Descarga de modelos entrenados

Los modelos se guardan en:

```
Mi unidad/New_ML_DL_models_stc_optimized/run_YYYYMMDD_HHMMSS/
```

Dentro de `artifacts/` se encuentran:

- `ensemble_knn.pkl`
- `random_forest.pkl`
- `cnn_lstm_attention.keras`
- `bilstm_attention.keras`
- `scaler.pkl`
- `metadata.json`

- `results_summary.csv`

Estos archivos deben copiarse a la carpeta `models/` del proyecto local.

5.7. Análisis espectral de señales (opcional)

El notebook `EMG_Spectral_Analysis.ipynb` permite:

- Analizar espectros de frecuencia por canal.
- Comparar movimientos SAFE vs RISK.
- Calcular energía por bandas de frecuencia.
- Generar mapas de calor y medidas de consistencia entre repeticiones.

Se configura de forma similar, ajustando únicamente la ruta de datos.

6. Interpretación de resultados

6.1. Tarjetas de resultados

Cada señal genera una tarjeta con:

- Color de fondo:
  - Rojo: movimiento de RIESGO detectado.
  - Verde: movimiento SEGURO.
- Mensaje principal: alerta o confirmación de seguridad.
- Nivel de confianza (en Sistema Dual).
- Probabilidades de clasificación (modo independiente).
- Estadísticas:
  - Total de ventanas.
  - Ventanas clasificadas como RISK.
  - Ventanas clasificadas como SAFE.
  - Porcentaje de riesgo.
- Gráfica de un canal EMG.
- Metadata (si está disponible): sujeto, movimiento, repetición.

6.2. Métricas de evaluación

Las métricas usadas incluyen:

- **Accuracy**.
- **Precision**.
- **Recall** (sensibilidad para la clase RISK).
- **F1-Score**.
- **AUC** (área bajo la curva ROC).

Para el problema de detección de movimientos de riesgo:

- El **recall** de la clase RISK es crítico (evitar falsos negativos).
- La **precision** también es importante (evitar demasiados falsos positivos).

6.3. Cómo actuar según los resultados

Si la señal se clasifica como RIESGO:

- No es un diagnóstico médico, sino una alerta preventiva.
- Revisar la calidad de la señal y el contexto del movimiento.
- Considerar acciones ergonómicas (cambios de postura, reducción de repetición, pausas activas).
- Si hay síntomas compatibles con STC (dolor, hormigueo, debilidad), acudir a valoración médica.

Si la señal se clasifica como SEGURO:

- El movimiento se considera de bajo riesgo según el modelo.
- Se recomienda mantener buenas prácticas ergonómicas.

En el Sistema Dual:

- Confianza *ALTA*: ambos modelos coinciden (SAFE o RISK).
- Confianza *MEDIA*: desacuerdo entre modelos; el sistema adopta la opción más conservadora (prioriza RISK).

6.4. Aviso legal

HeClaMoSTC es una herramienta de investigación y apoyo, no un dispositivo médico regulado. No sustituye:

- Diagnóstico médico.
- Evaluación clínica.
- Pruebas de laboratorio o estudios neurofisiológicos clínicos.

7. Solución de problemas

7.1. Problemas frecuentes en la aplicación

Problema	**Causa posible**	**Solución**
El servidor no inicia	Puerto 5000 ocupado	Cambiar el puerto en `server.py` (por ejemplo, 5001).
`ModuleNotFoundError`	Dependencias no instaladas	Ejecutar `pip install -r requirements.txt`.
Sistema Dual no disponible	Falta `metadata.json`	Copiar `metadata.json` a la carpeta `models/`.
Modelos DL no cargan	Formato incorrecto (.h5 en lugar de .keras)	Re-entrenar con el notebook actualizado que guarda en formato `.keras`.
Señales no aparecen en lista	Ruta configurada incorrectamente	Verificar y ajustar la ruta de `signals/` en `server.py`.
Error al clasificar	Archivo `.mat` corrupto o sin variable `emg`	Revisar el archivo, comprobar estructura y dimensiones.
Todas las predicciones iguales	Problema con `scaler.pkl`	Asegurarse de usar el `scaler.pkl` generado junto con los modelos.
Interfaz no carga	Faltan archivos `index.html` o `app.js`	Copiarlos a la carpeta `frontend/`.
Botón de clasificar deshabilitado	Faltan selección de modelo o señales	Escoger modo, modelo (si aplica) y al menos una señal.

7.2. Problemas con Google Colab

Error al montar Google Drive:

- Verificar que se ha dado permiso correctamente.
- Probar en ventana de incógnito.
- Volver a ejecutar el montaje.

Out of Memory:

- Reducir `BATCH_SIZE`.
- Usar menos sujetos (`USE_ALL_SUBJECTS = False`).
- Entrenar modelos por separado.

Entrenamiento muy lento:

- Verificar que la GPU esté activa con `!nvidia-smi`.
- Reducir número de épocas.
- Entrenar primero solo los modelos ML.

Runtime desconectado o sesión reiniciada:

- Mantener actividad mínima en la pestaña.
- Aprovechar que los modelos se guardan en disco al finalizar cada bloque.

Apéndice A: Estructura técnica del pipeline

El pipeline principal de HeClaMoSTC aplica las siguientes etapas:

1. Carga del archivo `.mat` y extracción de `emg`.
2. Filtrado pasa-banda Butterworth (20–450 Hz).
3. Filtro notch a 50 Hz.
4. Normalización Z-score por canal.
5. Segmentación en ventanas de 500 ms con 25 % de solapamiento.
6. Extracción de características (para modelos ML).
7. Preparación de secuencias (para modelos DL).
8. Clasificación por ventana.
9. Agregación de resultados por señal.
10. Decisión final (modo independiente o Sistema Dual).

Apéndice B: Especificaciones de modelos

Modelo Ensemble Subspace KNN

- Tipo: K-Nearest Neighbors en ensamble (bagging).
- Vecinos: $k = 7$.
- Número de clasificadores: 30.
- Ventajas: robustez al ruido, buen desempeño en datasets medianos.

Modelo Random Forest

- Tipo: ensamble de árboles de decisión (CART).
- Número de árboles: 200.
- Máxima profundidad: 30.
- Aporta rapidez en inferencia y capacidad de manejar datos desbalanceados con técnicas de sobremuestreo.

Modelo CNN + LSTM + Attention

Arquitectura híbrida que combina:

- Capas convolucionales 1D para capturar patrones locales.
- Capas LSTM bidireccionales para dependencias temporales.
- Mecanismo de atención para ponderar regiones temporales relevantes.
- Capa de salida sigmoide para probabilidad binaria RISK/SAFE.

Modelo BiLSTM + Attention

Similar al anterior pero con énfasis en el modelado secuencial usando BiLSTM y atención, sin convoluciones iniciales, lo que le da otra forma de capturar la dinámica temporal.

Bibliografía

[1] Ministerio de la Protección Social de Colombia, "Guía de atención integral basada en la evidencia para desórdenes musculoesqueléticos (dme) relacionados con movimientos repetitivos de miembros superiores (síndrome de túnel carpiano, epicondilitis y enfermedad de de quervain) (gati-dme)," https://www.epssura.com/guias/guias_mmss.pdf, 2020, accedido: 2025-05-08.

[2] K. Osiak, P. Elnazir, J. A. Walocha, and A. Pasternak, "Carpal tunnel syndrome: state-of-the-art review," *Folia Morphologica*, vol. 81, no. 4, pp. 851–862, Nov. 2021.

[3] A. Genova, O. Dix, A. Saefan, M. Thakur, and A. Hassan, "Carpal tunnel syndrome: A review of literature," *Cureus*, vol. 12, no. 3, Mar. 2020.

[4] F. P. Sánchez, O. Garcia, and M. I. R. Casallas, "Carga de la enfermedad atribuible al síndrome de túnel del carpo en la población trabajadora colombiana: Una aproximación a los costos indirectos de una enfermedad," *Value in Health Regional Issues*, vol. 2, no. 3, pp. 381–386, Dec. 2013.

[5] B. M. A. van Rijn, B. W. Koes, and A. Burdorf, "Associations between work-related factors and the carpal tunnel syndrome—a systematic review," *Scandinavian Journal of Work Environment & Health*, vol. 35, no. 1, pp. 19–36, Jan. 2009.

[6] G. Spahn, J. Wollny, B. Hartmann, R. Schiele, and G. Hofmann, "Metaanalyse zur bestimmung von risikofaktoren für das karpaltunnelsyndrom (kts) teil i. allgemeine risikofaktoren," *Zeitschrift für Orthopädie und Unfallchirurgie*, vol. 150, no. 05, pp. 503–515, Oct. 2012.

[7] M. H. Pourmemari and R. Shiri, "Diabetes as a risk factor for carpal tunnel syndrome: a systematic review and meta-analysis," *Diabetic Medicine*, vol. 33, no. 1, pp. 10–16, Jul. 2015.

[8] S. R, "Arthritis as a risk factor for carpal tunnel syndrome: a meta-analysis," *Scandinavian Journal of Rheumatology*, 2016.

[9] K. T. Palmer, E. C. Harris, and D. Coggon, "Carpal tunnel syndrome and its relation to occupation: a systematic literature review," *Occupational Medicine*, vol. 57, no. 1, pp. 57–66, Aug. 2006.

[10] A. Hassan, A. Beumer, and P. van, "Work-relatedness of carpal tunnel syndrome: Systematic review including meta-analysis and grade," *Health Science Reports*, vol. 5, no. 6, Nov. 2022.

[11] A. Lavé, R. Gondar, A. K. Demetriades, and T. R. Meling, "Ergonomics and musculoskeletal disorders in neurosurgery: a systematic review," *Acta Neurochirurgica*, vol. 162, no. 9, pp. 2213–2220, Jul. 2020.

[12] M. El-Helaly, H. H. Balkhy, and L. Vallenius, "Carpal tunnel syndrome among laboratory technicians in relation to personal and ergonomic factors at work," *Journal of Occupational Health*, vol. 59, no. 6, pp. 513–520, Aug. 2017.

[13] G. S. Jorge, "Estudio de la influencia de la ansiedad, la depresión y el distrés psicológico preoperatorios en pacientes intervenidos quirúrgicamente por presentar síndrome del túnel carpiano," Ph.D. dissertation, Universidad de Alcalá, 2021. [Online]. Available: http://hdl.handle.net/10017/50896

[14] F. P. Sánchez, "Calidad de vida en el trabajador con síndrome del túnel del carpo," *Revista Colombiana de Salud Ocupacional*, vol. 5, no. 1, pp. 13–18, Jun. 2015.

[15] Universidad Nacional de Colombia, "Cost of carpal tunnel syndrome diminishes with corticoids," https://agenciadenoticias.unal.edu.co/detalle/cost-of-carpal-tunnel-syndrome-diminishes-with-corticoids, Nov. 2014, accessed: 2025-04-17.

[16] E. del Barrio, J. Durán-Cantolla, S. Quintana, and E. Martínez-Vila, "Tratamiento conservador en pacientes con síndrome del túnel carpiano con intensidad leve o moderada. revisión sistemática," *Neurología*, vol. 33, no. 9, pp. 590–601, Jul. 2016.

[17] Universidad del Rosario, "Costo-utilidad de intervenciones en pacientes con síndrome del túnel carpiano atendidos en un centro de alta complejidad en Cali, Colombia," *Revista Universidad del Rosario*, 2015, accessed: 2025-04-17.

[18] M. V. S, H. L. A, Y. Fouad, and M. E. M. Soudagar, "Electromyography signal based hand gesture classification system using hilbert huang transform and deep neural networks," *Heliyon*, vol. 10, no. 11, p. e32211, Jun. 2024.

[19] M. Yetiş, H. Kocaman, M. Canlı, H. Yıldırım, A. Yetiş, and İ. Ceylan, "Carpal tunnel syndrome prediction with machine learning algorithms using anthropometric and strength-based measurement," *PLOS ONE*, vol. 19, no. 4, p. e0300044, Apr. 2024.

[20] Z.-M. Li and D. B. Jordan, "Carpal tunnel mechanics and its relevance to carpal tunnel syndrome," *Human Movement Science*, vol. 87, pp. 103 044–103 044, Nov. 2022.

[21] R. Martinek *et al.*, "Advanced bioelectrical signal processing methods: Past, present, and future approach—part iii: Other biosignals," *Sensors*, vol. 21, no. 18, pp. 6064–6064, Sep. 2021.

[22] C. L. Kok, C. K. Ho, F. K. Tan, and Y. Y. Koh, "Machine learning-based feature extraction and classification of emg signals for intuitive prosthetic control," *Applied Sciences*, vol. 14, no. 13, pp. 5784–5784, Jul. 2024.

[23] M. Boyer, L. Bouyer, J.-S. Roy, and A. Campeau-Lecours, "Reducing noise, artifacts and interference in single-channel emg signals: A review," *Sensors*, vol. 23, no. 6, pp. 2927–2927, Mar. 2023.

[24] A. Phinyomark and E. Scheme, "Emg pattern recognition in the era of big data and deep learning," *Big Data and Cognitive Computing*, vol. 2, no. 3, pp. 21–21, Aug. 2018.

[25] B. Mahesh, "(pdf) machine learning algorithms -a review," https://doi.org/10.21275//ART20203995, 2019, researchGate.

[26] I. Costa, "Tipos y aplicaciones del machine learning | artículo kaizen," https://kaizen.com/es/insights-es/tipos-aplicaciones-machine-learning/, Jul. 2024, accessed: 2025-05-24.

[27] J. Waring, C. Lindvall, and R. Umeton, "Automated machine learning: Review of the state-of-the-art and opportunities for healthcare," *Artificial Intelligence in Medicine*, vol. 104, p. 101822, Apr. 2020.

[28] E. Kavlakoglu, "Random forest," *IBM*, Oct. 2021, accessed: Nov. 10, 2025. [Online]. Available: https://www.ibm.com/think/topics/random-forest

[29] H. Abdel-Jaber, D. Devassy, A. A. Salam, L. Hidaytallah, and M. El-Amir, "A review of deep learning algorithms and their applications in healthcare," *Algorithms*, vol. 15, no. 2, pp. 71–71, Feb. 2022.

[30] G. J. Rani, "Ieee xplore full-text pdf," https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=10254204, 2025, accessed: 2025-05-25.

[31] M. Atzori *et al.*, "Electromyography data for non-invasive naturally-controlled robotic hand prostheses," *Scientific Data*, vol. 1, no. 1, Dec. 2014.

[32] Z. Ding, C. Yang, Z. Tian, C. Yi, Y. Fu, and F. Jiang, "semg-based gesture recognition with convolution neural networks," *Sustainability*, vol. 10, no. 6, p. 1865, Jun. 2018.

[33] K. Rezaee, S. F. Khavari, M. Ansari, F. Zare, and M. Hossein, "Hand gestures classification of semg signals based on bilstm-metaheuristic optimization and hybrid u-net-mobilenetv2 encoder architecture," *Scientific Reports*, vol. 14, no. 1, pp. 31 257–31 257, Dec. 2024.

[34] D. Xiong, D. Zhang, X. Zhao, and Y. Zhao, "Deep learning for emg-based human-machine interaction: A review," *IEEE/CAA Journal of Automatica Sinica*, vol. 8, no. 3, pp. 512–533, Feb. 2021.

[35] I. Sommerville, *Software Engineering*, 10th ed. Pearson, 2015.

[36] R. S. Pressman and B. R. Maxim, *Software Engineering: A Practitioner's Approach*, 8th ed. McGraw-Hill Education, 2014.

[37] C. Shearer, "The crisp-dm model: The new blueprint for data mining," *Journal of Data Warehousing*, vol. 5, no. 4, pp. 13–22, 2000.

[38] R. Wirth and J. Hipp, "Crisp-dm: Towards a standard process model for data mining," in *Proceedings of the 4th International Conference on the Practical Applications of Knowledge Discovery and Data Mining*, 2000, pp. 29–39.

[39] U. A. d. M. Instituto de Ingeniería del Conocimiento (IIC). (2021, noviembre) La metodología crisp-dm en ciencia de datos. Consultado en septiembre de 2025. [Online]. Available: https://www.iic.uam.es/innovacion/metodologia-crisp-dm-ciencia-de-datos/

[40] J. Chang, A. Phinyomark, and E. Scheme, "Assessment of emg benchmark data for gesture recognition using the ninapro database," in *Proceedings of the 42nd Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, Jul. 2020.