ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA DE SISTEMAS

PLATAFORMA PARA DESARROLLO Y EVALUACIÓN DE MODELOS DE RECONOCIMIENTO DE GESTOS DE LA MANO.

MODELO DE RECONOCIMIENTO DE 11 GESTOS DE LA MANO USANDO DEEP LEARNING Y CELDAS DE MEMORIA DESARROLLADO EN PYTHON

TRABAJO DE INTEGRACIÓN CURRICULAR PRESENTADO COMO REQUISITO PARA LA OBTENCIÓN DEL TÍTULO DE INGENIERO DE SOFTWARE

MARCO ANDRÉS SALAZAR FRANCO

marco.salazar02@epn.edu.ec

DIRECTOR: LORENA ISABEL BARONA LÓPEZ, PHD

lorena.barona@epn.edu.ec

DMQ, septiembre 2023

CERTIFICACIONES

Yo, Marco Andrés Salazar Franco declaro que el trabajo de integración curricular aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

_____

MARCO ANDRÉS SALAZAR FRANCO

Certifico que el presente trabajo de integración curricular fue desarrollado por Marco Andrés Salazar Franco, bajo mi supervisión.

_____

LORENA ISABEL BARONA LÓPEZ

DIRECTORA

DECLARACIÓN DE AUTORÍA

A través de la presente declaración, afirmamos que el trabajo de integración curricular aquí descrito, así como el (los) producto(s) resultante(s) del mismo, son públicos y estarán a disposición de la comunidad a través del repositorio institucional de la Escuela Politécnica Nacional; sin embargo, la titularidad de los derechos patrimoniales nos corresponde a los autores que hemos contribuido en el desarrollo del presente trabajo; observando para el efecto las disposiciones establecidas por el órgano competente en propiedad intelectual, la normativa interna y demás normas.

MARCO ANDRÉS SALAZAR FRANCO

LORENA ISABEL BARONA LÓPEZ

DEDICATORIA

Dedico este trabajo a mis padres que siempre me han brindado todas las facilidades necesarias para desarrollarme en el ámbito académico.

A mis amigos David, Erick, Glenn, Steven y Joel, los cuales en todo el proceso académico me han acompañado y con los momentos que vivimos pude tener la paz y las fuerzas para continuar con este camino.

AGRADECIMIENTO

Quiero expresar mi agradecimiento a todos los profesores que me brindaron su conocimiento para realizar este trabajo, en especial a los doctores Lorena Barona, Marco Benalcázar y Leonardo Valdivieso, por su guía en todo este proceso

De igual forma agradecer al Laboratorio de Investigación en Inteligencia y Visión Artificial “Alan Turing”, por apoyarme en la realización de este trabajo, en especial a Jonathan Zea por sus consejos y brindarme su conocimiento.

Finalmente, agradecer a Kevin Ramos que con su aporte fue posible terminar la integración de este componente y obtener los mejores resultados posibles.

ÍNDICE DE CONTENIDO

RESUMEN

Hoy en día, los progresos tecnológicos y su implementación en áreas como la automatización, la asistencia virtual y la interacción entre humanos y máquinas están en constante evolución. Sin embargo, a medida que la inteligencia artificial (IA) sigue expandiendo sus horizontes, también surgen desafíos significativos relacionados con la replicabilidad de los modelos en entornos reales. El presente componente de investigación se centra en el desarrollo de un modelo de Deep Learning basado en redes neuronales convolucionales (CNN) y redes con celdas de memoria (LSTM) para la clasificación y reconocimiento en tiempo real de 11 gestos de la mano. El modelo hace uso de señales electromiográficas (EMG) y de cuaternión (IMU) como entrada. En este documento, profundizaremos en el proceso de creación de un modelo utilizando la metodología CRISP-ML(Q) para el aprendizaje automático. Siguiendo cada etapa de esta metodología, incluyendo la adquisición de señales, el preprocesamiento, la extracción de características, la clasificación y el post procesamiento. Con esta metodología se obtuvieron resultados como un 60.61% de exactitud de clasificación del modelo y un 21.18% de exactitud de reconocimiento, para ejemplos de validación, haciendo uso de una aplicación para evaluación de modelos de reconocimiento de gestos de la mano.

PALABRAS CLAVE: Deep Learning, convolución, celdas de memoria, redes neuronales, reconocimiento y clasificación de gestos de la mano, electromiografía, cuaternión.

ABSTRACT

Nowadays, technological advancements and their implementation in areas such as automation, virtual assistance, and human-machine interaction are constantly evolving. However, as artificial intelligence (AI) continues to expand its horizons, significant challenges related to model replicability in real-world settings also arise. The present research component focuses on the development of a Deep Learning model based on Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks for real-time classification and recognition of 11 hand gestures. The model utilizes electromyographic (EMG) and quaternion (IMU) signals as input. In this document, we delve into the process of creating a model using the CRISP-ML(Q) methodology for machine learning. Following each stage of this methodology, including signal acquisition, pre-processing, feature extraction, classification, and post-processing. Using this methodology, results were achieved, including a classification accuracy of 60.61% for the model and a recognition accuracy of 21.18%, for validation examples, using an application for evaluating hand gesture recognition models.

KEYWORDS: Deep Learning, convolution, memory cells, neural networks, hand gesture recognition and classification, electromyography, quaternion.

1 DESCRIPCIÓN DEL COMPONENTE DESARROLLADO

En el presente componente se ha desarrollado un modelo de Deep Learning que se basa en una arquitectura combinada de red convolucional (CNN) y una red con celdas de memoria (LSTM) para la clasificación y reconocimiento de 11 gestos de la mano (5 gestos estáticos, 6 gestos dinámicos y un gesto de relajación). Específicamente, los gestos utilizados son: palma de la mano a la izquierda (wave in), palma de la mano a la derecha (wave out), puño (fist), apertura de la palma (open), juntar dos dedos dos veces (pinch), brazo hacia arriba (up), brazo hacia abajo (down), brazo a la izquierda (left), brazo a la derecha (right), brazo hacia adelante (forward) y brazo hacia atrás (backward).

El reconocimiento de gestos se divide en 5 etapas principales: adquisición de la señal, pre procesamiento, extracción de características, clasificación y post procesamiento. En la etapa de adquisición se recolectan 2 tipos de señales: EMGs e IMUs. En el pre procesamiento, los datos se transformarán en formato JSON para su uso en Python; asimismo, se aplicarán procesos de limpieza, filtrado y normalización para obtener una señal sin ruido. Para la extracción de características se utiliza una red neuronal con el fin de obtener parámetros que permitan diferenciar características especiales para cada tipo de gesto.

Finalmente, la función de post procesamiento examina las etiquetas asignadas durante el proceso de clasificación y determina la clase para la secuencia completa de gestos. Esta función reemplaza las etiquetas de gestos individuales si difieren de la clase de la secuencia general, brindando así una visión coherente del conjunto de gestos en su totalidad.

Al terminar el entrenamiento de los modelos se utilizarán datos de prueba que no han sido usados anteriormente para evaluar el rendimiento del modelo, con la ayuda de la plataforma de evaluación con el objetivo de entender el rendimiento del modelo frente a datos no conocidos.

1.1 Objetivo general

Clasificar y reconocer once gestos de la mano utilizando un modelo de inteligencia artificial basado en redes neuronales convolucionales y celdas de memoria implementado en Python.

1.2 Objetivos específicos

- Examinar la literatura actual sobre las últimas tendencias en el diseño de arquitecturas de redes neuronales que hacen uso de señales electromiográficas junto con unidades de medición inercial, para el reconocimiento y clasificación de gestos de la mano.
- Organizar los datos de las señales EMG e IMU en formato .JSON.
- Crear un modelo de reconocimiento y clasificación par gestos de la mano con un modelo CNN-LSTM, usando el marco metodológico CRISP-ML(Q) para el desarrollo de modelos
- Evaluar el modelo desarrollado en términos de eficiencia en clasificación y reconocimiento.

1.3 Alcance

El campo de la inteligencia artificial y el aprendizaje automático está experimentando actualmente un notable aumento en el número de proyectos que se llevan a cabo, y sus aplicaciones pueden llegar a ser de vital importancia para el desarrollo tecnológico de la humanidad. La inteligencia artificial (IA) tiene una amplia variedad de aplicaciones en diversos campos y sectores, algunas de las cuales son: automatización de procesos empresariales, asistencia virtual, análisis de datos, automatización industrial, automóviles autónomos, cuidado de la salud, y muchos otros ejemplos. De igual manera, el reconocimiento de gestos manuales (HGR, por sus siglas en inglés) es importante porque permite a las máquinas responder a los gestos de las manos de los usuarios. Esto es relevante en el contexto de la interfaz hombre-máquina, donde el reconocimiento de gestos puede permitir a los usuarios interactuar con las máquinas de una manera más natural e intuitiva.

El presente componente consiste en la implementación, en Python, de un modelo de reconocimiento de 11 gestos de la mano. Como entrada del modelo, se recibirán archivos “JSON1” con los datos de señales EMG y señales IMU por cada usuario. Como salida del modelo se generará un archivo “JSON2” para guardar las predicciones obtenidas del modelo HGR. Cabe recalcar que, como parte de este componente se definirá el formato

JSON 1. En la Figura 1 se ilustra los gestos que el modelo de redes neuronales convolucionales y celdas de memoria reconocerá.

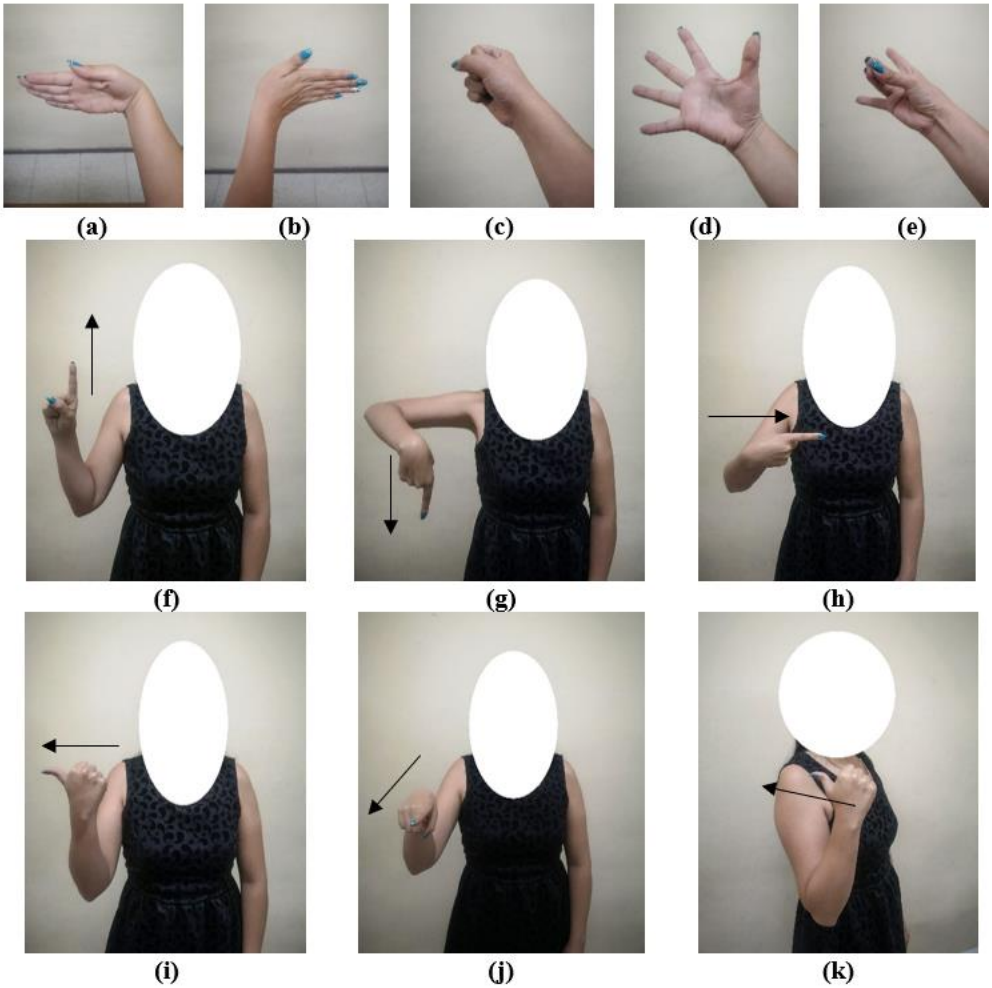


Figura 1 Gestos que se plantea reconocer en este proyecto. (a) palma de la mano a la izquierda (wave in), (b) palma de la mano a la derecha (wave out), (c) puño (fist), (d) apertura de la palma (open), (e) juntar dos dedos dos veces (pinch), (f) brazo hacia arriba (up), (g) brazo hacia abajo (down), (h) brazo a la izquierda (left), (i) brazo a la derecha (right), (j) brazo hacia adelante (forward) y (k) brazo hacia atrás (backward).

En particular, el desarrollo de este componente utilizará el modelo de proceso CRISP-ML(Q) [3] como una metodología de desarrollo del modelo de clasificación de gestos. La metodología CRISP-ML(Q) es un proceso dinámico y flexible que permite mejorar constantemente la precisión de los modelos de machine learning. En esta metodología se consideran las siguientes fases:

Comprensión del problema: En esta fase, se identifica y se define el problema que se quiere resolver y los datos relevantes que se requieren para hacerlo.

1. Preparación de datos: Se limpian, integran y transforman los datos para que estén listos para el análisis.
2. Selección de modelos: En esta fase, se evalúan diferentes modelos de machine learning para determinar cuál es el más adecuado para el problema en cuestión.
3. Entrenamiento: En este paso, se ajustan los parámetros y se entrena el modelo seleccionado.
4. Evaluación: Se evalúa el rendimiento del modelo para determinar su eficacia en la solución del problema.
5. Implementación: En esta fase, se integra el modelo en una aplicación o sistema y se prueba en un entorno real.
6. Mantenimiento: En este último paso, se monitorean y actualizan los modelos para asegurarse de que sigan siendo precisos y relevantes.

Estas fases se realizan de manera iterativa, es decir, se pueden volver a revisar y mejorar en cualquier momento.

1.4 Marco teórico

Señales EMG e IMU

El reconocimiento de gestos de la mano tiene una gran variedad de aplicaciones, como por ejemplo en dispositivos de interacción de humano-robot [1]. En particular, el reconocimiento de gestos mediante señales electromiográficas (EMGs) y las unidades de medida inerciales (IMUs) necesitan modelos de clasificación que permitan determinar a qué clase pertenece un movimiento y su instante de ocurrencia [2].

La electromiografía (EMG) y los estudios de conducción nerviosa son pruebas que miden la actividad eléctrica de los músculos y nervios [4]. Es decir, permiten entender como los músculos y nervios se comportan a nivel eléctrico al momento en que se realiza una acción en específico. Por lo tanto, por cada movimiento se registra una señal diferente y se espera que la señal sea igual o muy parecida si se realiza otra vez el mismo movimiento.

El tener esta información en forma de señales permite tener un gran abanico de posibles aplicaciones. Por ejemplo, las señales EMG se pueden utilizar para la determinación del tiempo de activación del músculo, la estimación de la fuerza producida por una contracción muscular y la obtención de información sobre la fatiga muscular [5]. En este caso en específico, las señales EMG ayudan a identificar que movimiento está realizando el musculo del que se obtiene la señal. El uso de señales EMG tiene diversas aplicaciones, algunas pueden ser:

- Prótesis de extremidades: Pueden ayudar a manejar prótesis de diferentes partes del cuerpo, permitiendo a personas que por algún problema médico no puedan utilizar una extremidad manejar una prótesis a voluntad [6].
- Rehabilitaciones: Las señales EMG pueden utilizarse en aplicaciones de diagnóstico y rehabilitación de pacientes con lesiones o trastornos neuromusculares [7].
- Interfaces humano-computador: Estas señales pueden controlar interfaces para dispositivos electrónicos, como juegos, computadoras y dispositivos móviles, así como cualquier otro tipo de aplicación de software [8].
- Investigación científica: se utilizan ampliamente en la investigación científica para estudiar la actividad muscular y comprender los mecanismos neuromusculares [4].

En resumen, las señales EMG son útiles en diferentes ámbitos tecnológicos.

Las unidades de medición inercial (IMU) miden la aceleración y la velocidad angular de un objeto en movimiento [9]. Normalmente estas mediciones se realizan con la ayuda de acelerómetros y giroscopios que miden la aceleración lineal y la velocidad angular, respectivamente. Al tener estos datos se puede conocer el movimiento que se está realizando, específicamente en este caso el movimiento del brazo (se puede deducir la orientación que tiene el brazo y como se dio el cambio de posición del brazo/mano). Gracias a estos dos tipos de señales podemos conocer qué movimiento de mano/brazo se está realizando, sin necesidad de estar presente, o en ciertos casos nos ayudan a entender el tipo de movimiento que se quiere realizar a pesar de no contar con la extremidad.

Redes CNN

Las redes neuronales convolucionales (CNN) son un tipo de red neuronal artificial que se utiliza principalmente para el procesamiento de imágenes y el reconocimiento de patrones [10]. Este tipo de redes sirven principalmente para reconocer objetos en imágenes. Este reconocimiento utiliza diferentes filtros para extraer características importantes dentro de una imagen y de esa forma conseguir los patrones que ayudan a reconocer lo que hay en una imagen. Una red neuronal convolucional (CNN) está conformada por una gran variedad de capas, las cuales se dividen principalmente en 3 tipos:

- Capa de convolución: Esta capa se encarga de aplicar filtros a la imagen de entrada para extraer características relevantes [11]. Las capas convolucionales se encargan de identificar patrones mediante la extracción características. Normalmente, se utilizan varias capas convolucionales, y se espera que cada capa obtenga un conjunto de características distinto del anterior.
- Capa de pooling: Esta capa se utiliza para reducir el tamaño de la imagen y disminuir la cantidad de parámetros que deben ser procesados por la red [11]. Este tipo de capa es de gran ayuda para la red ya que al reducir el tamaño de la imagen de entrada y al aplicar capas convolucionales a imágenes más pequeñas obtiene características cada vez más generales de las imágenes.
- Capa completamente conectada: Esta capa se utiliza para clasificar la imagen en diferentes categorías [11]. Esta capa es normalmente la capa final de la red neuronal y es la encargada de etiquetar la imagen según las clases que tenga la red.

Una red neuronal convolucional se basa en juntar los 3 tipos de capas, especialmente las capas convolucionales y de pooling, para formar una red profunda que va a depender de la complejidad del problema y las características que se esperen obtener de las imágenes en cuestión. En la Figura 2 se puede ver un ejemplo de una red convolucional, en este ejemplo de red se aplican capas convolucionales con un diferente número de filtros para después aplicar capas de pooling y reducir el tamaño de la imagen.

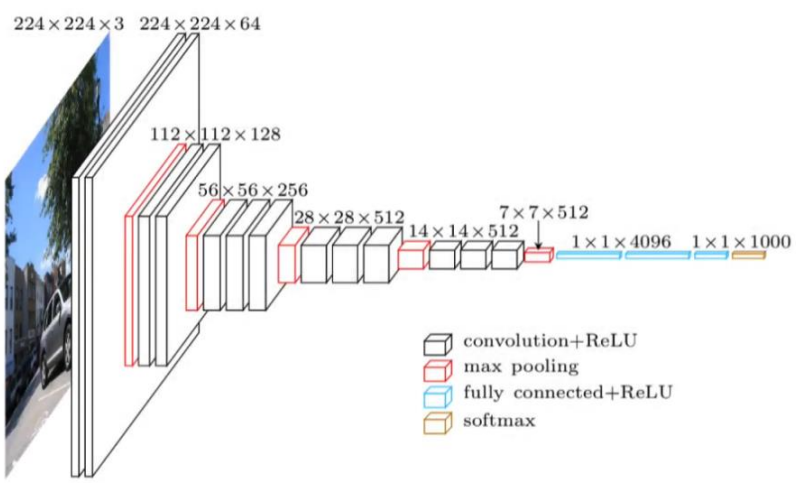


Figura 2 Red convolucional

Espectrogramas

Las redes convolucionales están específicamente desarrolladas para utilizar imágenes a las que se aplican ciertos filtros, pero para el reconocimiento de gestos de las manos se tiene señales de dos tipos: EMG e IMU. Por lo tanto, es necesario transformar las señales a espectrogramas.

Un espectrograma es una representación bilineal donde la potencia espectral de una señal no estacionaria es representada tanto en tiempo como en frecuencia [12]. Esta es una representación visual en donde cada punto en el espectrograma se convierte en un píxel en la imagen, donde la intensidad del píxel representa la potencia espectral en esa ubicación [13]. Por tanto, si se transforma la señal EMG e IMU a un espectrograma se tiene una señal representada como una imagen para la red convolucional.

Redes LSTM

Las redes neuronales recurrentes (RNN por sus siglas en inglés) son aquellas redes en las que la información no fluye en un único sentido (de la entrada a la salida), sino que disponen de conexiones hacia atrás, permitiendo así una retroalimentación [14]. Las redes LSTM son el tipo de red recurrente más común en la actualidad.

Las redes LSTM tienen una estructura similar a las redes recurrentes solamente que la operación de propagación de la información difiere de las recurrentes. En una red LSTM, hay tres tipos de compuertas: la compuerta de entrada, la compuerta de olvido y la compuerta de salida [15]. Gracias a estas compuertas cuando la información pasa a través de ella, la operación decide qué información procesar más y qué información debe dejar ir [16]. Por lo tanto, la red es capaz de recordar secuencias anteriores con una memoria de largo plazo, lo que ajusta los parámetros de una manera más eficiente.

2 METODOLOGÍA

La metodología CRISP-ML(Q) es usada durante el desarrollo del modelo de gestión, por lo tanto, se aplican cada una de sus etapas para obtener el mejor modelo posible. En la Figura 3 se muestra las diferentes etapas que tiene la metodología y la forma en la que se ejecutan estas etapas.

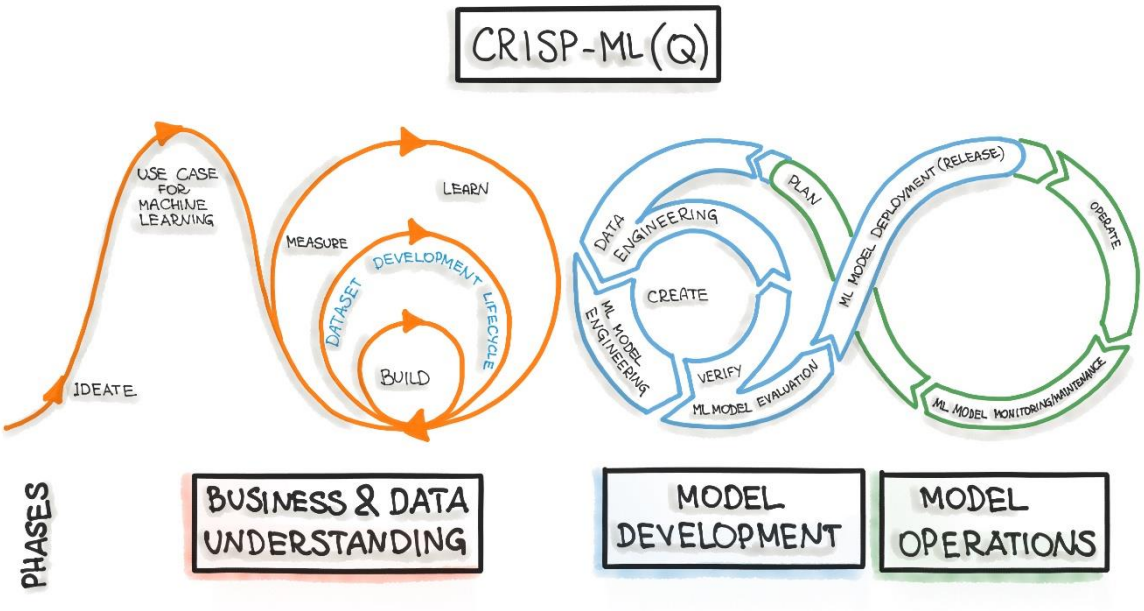


Figura 3 Etapas de la metodología CRISP-ML(Q)

En las siguientes secciones se describe como se ejecutaron cada una de las etapas de la metodología, que decisiones se tomaron y como se afrontaron las tareas necesarias para cumplir una etapa.

2.1 Preparación de datos

En la etapa de preparación de datos se ejecutan varias tareas, las cuales permiten tener los datos listos para ingresarlos al modelo para luego entrenarlo. También se establece una definición de la dimensionalidad de los datos de entrada, de modo que cuando se disponga de nuevos datos para la predicción, puedan adaptarse a la forma de entrada del modelo pre entrenado.

Primera fase: Obtención de datos y cambio de formato

Los datos utilizados en esta fase se encuentran en el Anexo I. Estos datos ya se encuentran estructurados y depurados de manera tal que su acceso es sencillo y sin muchas complicaciones. Es importante tomar en cuenta que los archivos de los datos se encuentran en un formato de archivo .mat. Este tipo de formato se usa en Matlab, y dado

que el proyecto se enfoca al uso de Python y JSON, la siguiente tarea a ejecutar es el cambio de formato de .mat a .json. En la Tabla 1 se muestra un resumen general de los datos más relevantes de los archivos.

Tabla 1 Propiedades y valores admitidos de un conjunto de muestras de un usuario

Propiedad	**Propiedad Valor admitido**
Número de muestras por usuario	360
Número de muestras por gesto	30
Intervalo de tiempo para el registro de una muestra	5 segundos
Frecuencia de muestreo del EMG	200 Hz (Myo) ó 500 Hz (GForce)
Frecuencia de muestreo de los cuaterniones	50 Hz
Número de canales del EMG	8
Forma de representación de los cuaterniones	Vector de coeficientes del cuaternión (w, x, y, z)
Tipo de dato de la etiqueta del gesto	Categorical, escogido del conjunto de clases compuesto por: *backward, forward, up, down, left, right, fist, pinch, waveIn, waveOut, relax*

Luego, se ejecuta un script que realiza el cambio de formato de los archivos (de .mat a .json). El principal objetivo de este cambio es que se mantenga la información del archivo original y que se utilice un formato más accesible para cualquier tipo de aplicación. Los archivos JSON serán utilizados en la siguiente fase.

Segunda fase: Entendiendo los datos

Para desarrollar el modelo de reconocimiento de gestos de la mano se utilizarán como entrada las señales EMG e IMU. Estas señales se recogerán de usuarios voluntarios, de los que se obtendrán 30 repeticiones de cada gesto. Este procedimiento se lleva a cabo con la ayuda de los dispositivos Myo Armband y GForce Pro. Estos son dispositivos portátiles que utilizan sensores de electromiografía (EMG) para detectar la actividad muscular y traducirla en señales de control para diversas aplicaciones [17]. Estos dispositivos cuentan con un total de 8 canales para la obtención de las señales. Cada dispositivo produce un vector discreto y normalizado [18] que representa la señal EMG medida para cada uno de sus 8 canales, reflejando el gesto realizado por el usuario. El

sensor proporciona un vector de rotación que capta los componentes w, x, y, z del cuaternión [18], compuesto por 4 canales que conforman la señal IMU de los gestos.

La frecuencia de muestreo de los dispositivos se puede configurar para que varíe. Se utilizaron dos tipos de dispositivos para obtener datos, que están configurados con 2 valores de frecuencia: i) 200Hz para Myo Armband y ii) 500Hz para GForce Pro. Con estas frecuencias configuradas se obtienen las señales EMG. Las muestras son tomadas durante 5 segundos por cada gesto, por lo tanto, el tamaño de las señales será diferentes dependiendo del dispositivo usado. Por otro lado, para las señales IMU se tiene una frecuencia estándar de 50Hz en ambos dispositivos lo que resulta en señales del mismo tamaño. Para el procesamiento de los datos, es necesario considerar que el presente trabajo realiza el reconocimiento de gestos. El reconocimiento requiere conocer el gesto que se realiza (clasificación) y en qué punto de la señal empieza y termina la realización del gesto.

Tercera fase: Procesamiento de datos

La Figura 4 muestra el flujo de trabajo de los modelos, con las etapas de procesamiento resaltados en verde. La primera etapa es la entrada al switch para clasificar el gesto como dinámico o estático. La segunda etapa es la entrada al reconocimiento del gesto por parte de los modelos CNN-LSTM.

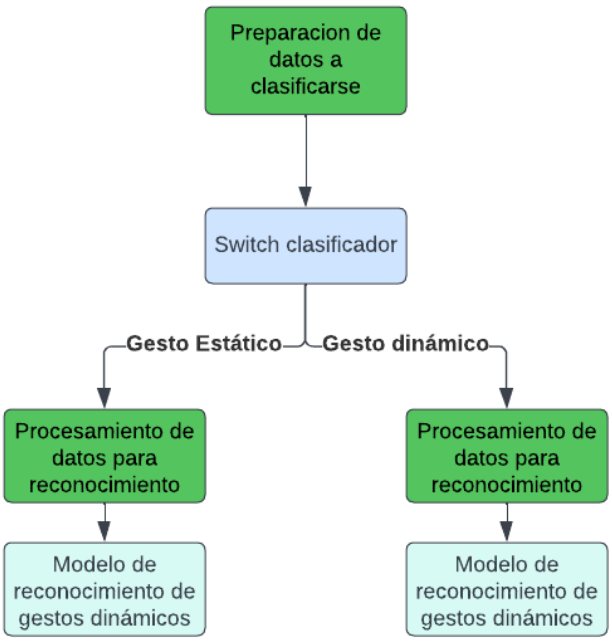


Figura 4 Flujo de trabajo de los modelos

En la primera etapa lo que se busca es generar un vector de características de los datos usando la función energía [19]. El vector mencionado es utilizado por el modelo inicial (switch), que es un clasificador logístico lineal. Este modelo se entrena utilizando el vector de características antes mencionado. En la Figura 5 se muestra el flujo de procesamiento de datos de la primera etapa.

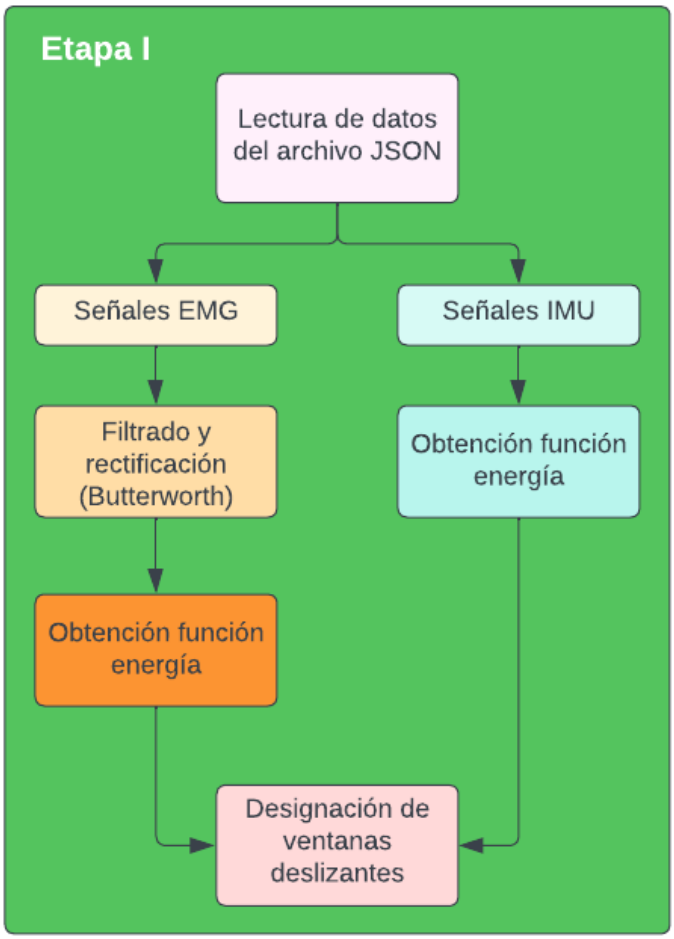


Figura 5 Primera etapa de procesamiento de datos

Para que la función energía obtenga resultados más precisos es necesario que en la señal EMG se realice el rectificado y filtrado usando un filtro Butterworth pasa bajos de quinto orden, de esta manera se logra eliminar el ruido de la señal.

Los sensores utilizados en este proyecto están equipados con 8 canales para captar señales de electromiografía (EMG). Así, por cada gesto realizado, se generan 8 señales EMG, además, se utilizan 4 canales para captar señales de la Unidad de Medición Inercial (IMU), lo que da lugar a 4 señales. En la fase inicial del procesamiento de datos, se calcula la función de energía de cada señal, lo que da como resultado un vector con 12 características.

La siguiente etapa de procesamiento de datos tiene como objetivo preparar los datos para su ingreso a los modelos de reconocimiento CNN-LSTM tanto de gestos dinámicos como estáticos. En esta fase se busca generar espectrogramas de las señales EMG e IMU para que se utilicen en el modelo. Con este objetivo, es necesario seguir una serie de pasos, que en su mayoría son los mismo para ambas señales con unas diferencias muy pequeñas. En la Figura 6 se muestra la etapa II de procesamiento de datos.

Figura 6 Segunda etapa de procesamiento de datos

Una vez cargadas las señales, se segmentan para obtener la parte de la señal en la que se realiza el gesto. Esto se hace con el fin de crear información útil que mejore la calidad y la eficacia de los posteriores procedimientos de preprocesamiento.

Una vez finalizada la segmentación de la señal, se aplica un proceso diferente en función de la rama sobre la que se trabaje. En la rama de la señal EMG, la señal se rectifica y se filtra, como se hizo en la etapa I de procesamiento de datos. En cambio, en la rama de la señal IMU, los datos se normalizan. Debido a su naturaleza, la señal de la IMU no presenta un ruido excesivo que deba corregirse.

Tras rectificar y filtrar la señal o normalizarla por defecto, se designa una ventana deslizante. Esta ventana deslizante tiene un tamaño fijo que es independiente de la señal o del dispositivo utilizado, ya que se define en tiempo (segundos). Sin embargo, como las frecuencias de las señales son diferentes, hay que hacer ciertos ajustes para mantener el

tiempo de la ventana. Estos cambios se realizan con el objetivo de tener ventanas del mismo tamaño independientemente de la señal utilizada.

Como paso final, las ventanas de señal se transforman en espectrogramas. Estos espectrogramas se generan utilizando las bibliotecas numpy de Python, y sus parámetros de generación se configuran para que las dimensiones de salida de las ventanas tengan las dimensiones (13,24). Estas dimensiones de los espectrogramas son independientes del tipo de señal, y los parámetros se configuran para que siempre se obtenga esta señal.

La dimensión descrita funciona para ventanas de una señal de un solo canal. Como hay 8 señales EMG y 4 señales IMU, el resultado total del procesamiento de la señal es un conjunto de espectrogramas con dimensiones de (13,24,8) o (13,24,4) según el tipo de señal. Por tanto, estas dimensiones constituyen la entrada de los modelos de reconocimiento, y para los nuevos ejemplos, ya sean de entrenamiento o de predicción, los datos deben tener estas dimensiones.

La figura 7 muestra el proceso completo de preparación de una señal EMG, basado en la descripción anterior. La única diferencia en el procesamiento de una señal IMU estará en el paso 7.b cuando sea necesario.

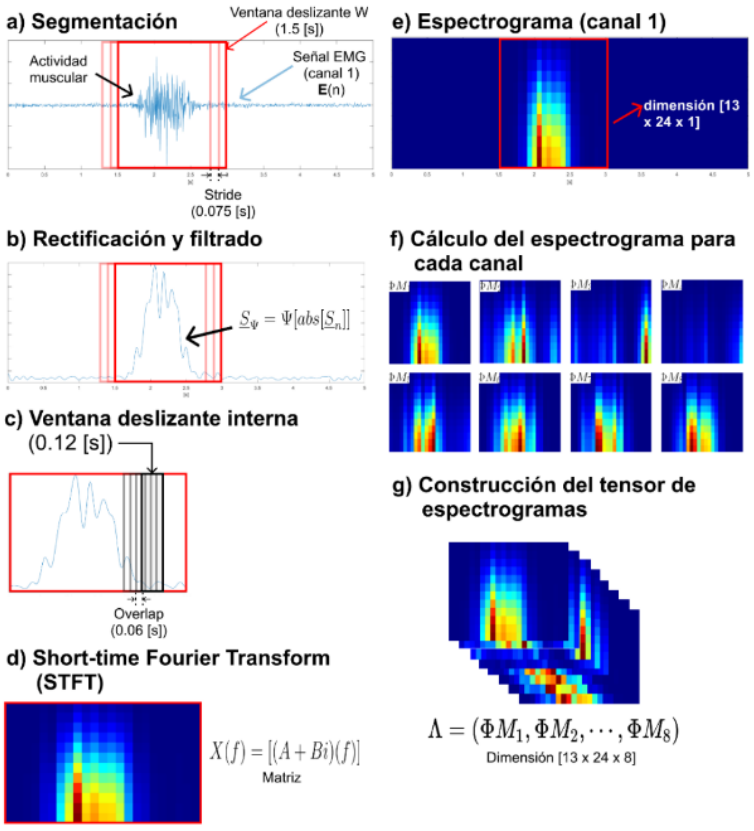


Figura 7 Selección, limpieza y construcción de datos para una señal EMG

El proceso de la etapa II de procesamiento de datos es un paso crítico en el análisis de datos que debe realizarse cada vez que se añaden nuevos datos. Este proceso constituye la base de la preparación de datos y es necesario tanto para entrenar con nuevos datos como para hacer predicciones.

2.2 Selección de modelos

Para llevar a cabo el reconocimiento de los gestos de la mano tanto dinámicos como estáticos, se proponen tres modelos de inteligencia artificial. Un modelo se utiliza para diferenciar entre un gesto estático y uno dinámico, mientras que los otros dos se utilizan para reconocer el gesto concreto que se está realizando.

Inicialmente, la clasificación de los gestos entre dinámicos y estáticos se realiza con un clasificador lineal logístico. Esta clasificación utiliza la función energía como entrada para clasificar entre tipos. En la Figura 8 se ilustra el proceso de funcionamiento del modelo de regresión logística para clasificar los tipos de gestos.

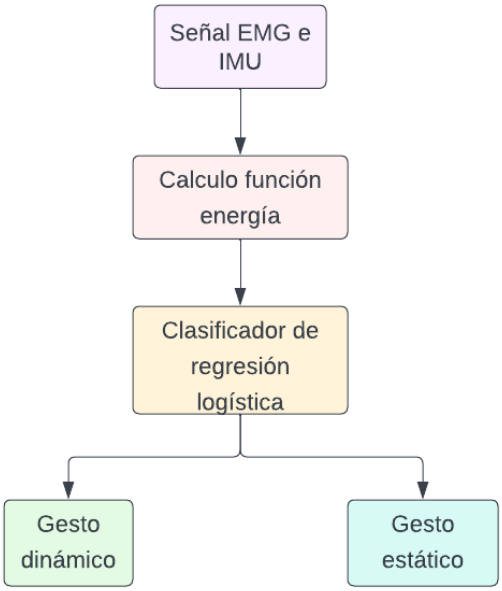


Figura 8 Proceso de clasificación del tipo de gesto con regresión logística

Ambos modelos de reconocimiento de gestos dinámicos y estáticos se basan en una arquitectura de red CNN-LSTM que utiliza el marco GoogLeNet [20]. Esta arquitectura emplea bloques de Inicio que consisten en múltiples capas convolucionales colocadas en paralelo a la entrada. Dependiendo de la rama, se puede aplicar o no la agrupación, y las salidas resultantes se concatenan en una única salida. La figura 9 ilustra la arquitectura utilizada en los bloques de Incepción.

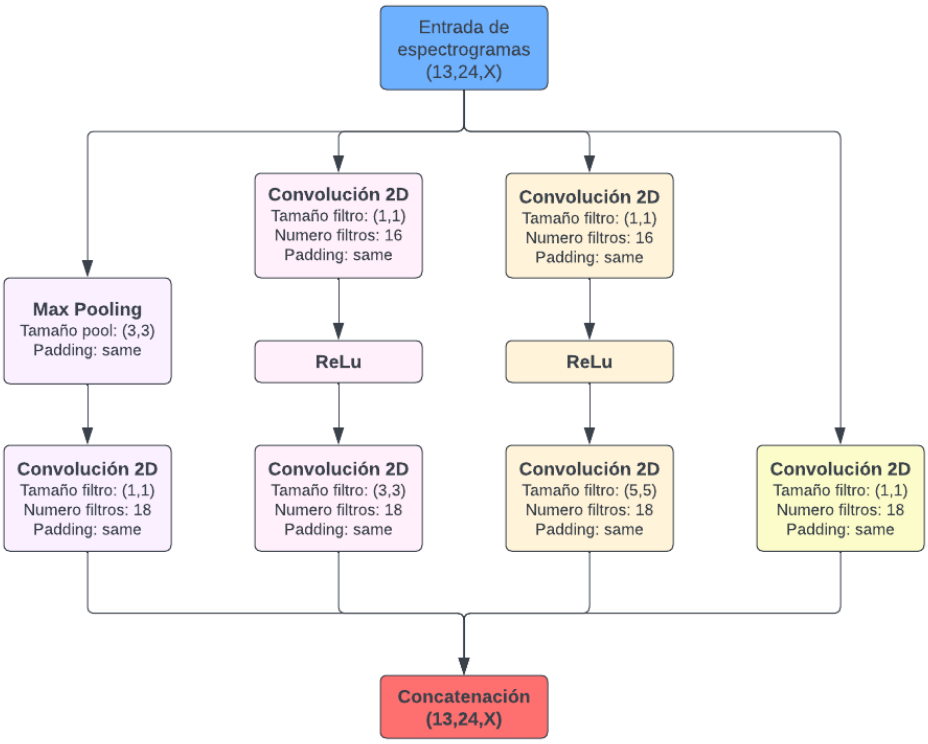


Figura 9 Bloque Inception utilizado en el modelo

En la Figura 9, se pueden distinguir las 4 ramas que tiene cada bloque. La primera rama de la izquierda consta de dos capas. La primera capa es una capa MaxPooling con un tamaño de pool de 3,3 y un stride "same". Esto garantiza que el tamaño 2D del espectrograma permanezca inalterado, y que sólo se actualicen los valores al máximo dentro del tamaño del pool. Tras la capa MaxPooling hay una capa Convolucional 2D con 18 filtros, cada uno con un tamaño de filtro de 1,1.

Por otro lado, la segunda rama del bloque comienza con una capa Convolucional 2D de 16 filtros, cada uno con un tamaño de 1,1 y padding "same". Le sigue una capa de activación ReLu y, por último, otra capa Convolucional 2D idéntica a la rama anterior, salvo que los 18 filtros tienen ahora un tamaño de 3,3.

La tercera rama tiene la misma estructura que la segunda, con la única diferencia de que en la segunda capa Convolucional se utiliza un tamaño de filtro de 5,5 y un padding "same" para mantener la relación de tamaño de salida.

Finalmente, la última rama del bloque consiste únicamente en una capa Convolucional 2D con 18 filtros y un tamaño de filtro de 1,1. La salida de cada capa se une mediante una capa de concatenación. Para que esta capa de concatenación funcione, la salida de

cada rama debe tener las mismas dimensiones en los dos primeros ejes. La salida de la capa de concatenación tiene unas dimensiones de (13,24,72), debido a que todas las ramas terminan con una capa convolucional de 18 filtros que, al concatenarse, dan como resultado un total de 72 filtros. Esto es independiente de si se trata de una señal EMG o IMU, ya que el número de filtros utilizados determina la tercera dimensión de las matrices, en lugar de la dimensión de entrada de los datos.

Se utiliza un bloque Inception para extraer características específicas de los espectrogramas, por lo que utilizar varios de estos bloques permite obtener características más relevantes. La Figura 10 muestra la arquitectura de extracción de características con estos bloques, y esta parte de los modelos es idéntica para los modelos de gestos dinámicos y estáticos.

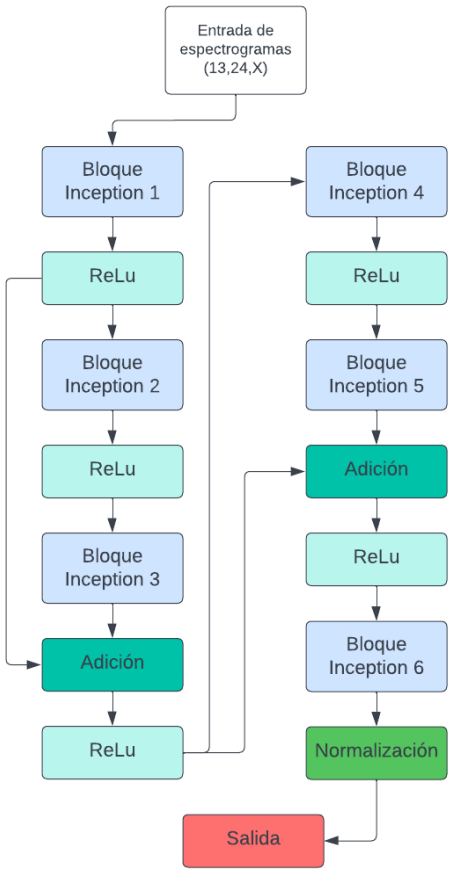


Figura 10 Arquitectura de extracción de características

La arquitectura descrita en la Figura 10 busca extraer características relevantes de los espectrogramas tanto de las señales EMG como IMU. Cada uno de los bloques Inception extrae ciertas características. Al terminar un bloque Inception entra a la capa ReLu para la rectificación, eso se repite repetidas veces. Existen 2 casos en los que entre estas

capas se introduce una capa de adición. Estas capas de Adición ayudan a que los valores de características externas no sean muy bajos mediante la suma de valores de salidas de capas anteriores. Esta es una de las principales razones por la que los bloques Inception dan como salida siempre estructuras con las mismas dimensiones ya que si los valores difieren entre sí sería imposible sumar.

En el último bloque se agrega una capa de Normalización, la cual sirve para que los valores que entren sean normalizados. En este caso se utiliza una normalización de batch, es decir que se normalizan los valores según el batch de datos que está pasando por el modelo, por lo tanto, este valor cambia según el batch size que se escoja para el entrenamiento. Finalmente, esta arquitectura se une a unas pocas capas más para completar el proceso, dentro de estas capas se encuentra la LSTM que es parte fundamental del modelo. En la Figura 11 se puede ver cómo sería la arquitectura del modelo para gestos estáticos y dinámicos.

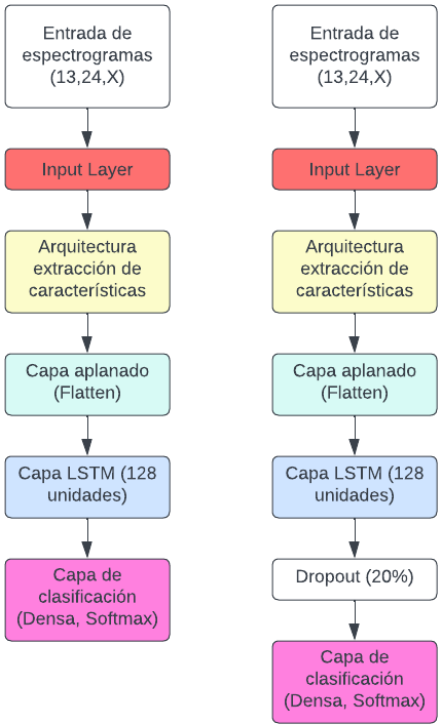


Figura 11 Modelos de reconocimiento de gestos estáticos y dinámicos

La salida de la arquitectura de extracción de características se conecta a una capa de aplanado, ya que capas posteriores a estas necesitan que los datos se encuentren en una dimensión y no en dos dimensiones. Después de aplanar los datos entran a la capa LSTM con 128 unidades ocultas, esta capa permite que el modelo aprenda datos

relevantes de la secuencia de entrada. Se tiene la capa Densa con 6 neuronas en el modelo de reconocimiento de gestos estáticos y con 7 neuronas para el modelo de gestos dinámicos. Este número depende de cuantas clases hay para clasificar, ya que esta capa se encarga precisamente de eso utilizando la función softmax. Existe una capa extra en el modelo de reconocimiento de gestos dinámicos, la cual es la capa de dropout. Esta capa se encarga de deshabilitar cierto número de neuronas durante el entrenamiento para evitar overfitting en el modelo.

Estos son los modelos iniciales planteados y con los que se empezará el proceso de entrenamiento. Es importante mencionar que los modelos se podrán modificar durante la fase de mantenimiento.

2.3 Entrenamiento

Para el entrenamiento de los modelos descritos es necesario configurar ciertos hiperparámetros los cuales afectan la forma en la los modelos se entrenan y pueden mejorar significativamente los resultados obtenidos.

Primer Modelo (Switch)

Para el modelo de regresión logística (switch) no se configuran hiperparámetros. Simplemente se ingresan los datos y, mediante el código se pre procesan los datos hasta obtener el vector de la función energía de las señales. Por lo tanto, el proceso de entrenamiento de este modelo solo se realiza una vez.

Segundo Modelo (Gestos Estáticos) y Tercer Modelo (Gestos dinámicos)

El modelo de reconocimiento de gestos estáticos y dinámicos (CNN-LSTM) al ser un tipo de red neuronal profunda consta de muchos hiperparámetros a configurar. Para entrenar estos modelos se seguirá una planificación de entrenamiento, en la cual modifican los valores de 3 hiperparámetros. En la Tabla 2 se muestra el proceso de modificación de hiperparámetros para obtener los mejores resultados.

Tabla 2 Hiperparámetros y sus valores a modificar

Batch Size	**Learning Rate**	**Num Epochs**
16	0.0005	10
32	0.001	15
64	0.002	20

En la Tabla 2 se muestran los valores que cambian en cada entrenamiento. Este entrenamiento empezará con los parámetros de la primera fila, modificando los valores de Batch Size hasta obtener los mejores resultados. Esta configuración se escogerá para cambiar los valores de Learning Rate y de igual forma el Número de Epochs. En la metodología CRISP-ML(Q) la fase de entrenamiento y evaluación están relacionadas, esto se debe a que cuando se termina un entrenamiento del modelo se evalúan los resultados basados en la función de perdida y la exactitud del modelo. Es decir que al terminar la parte de evaluación inicia una nueva de entrenamiento hasta tener los mejores resultados.

En el Anexo II se especifican los valores de los hiperparámetros escogidos para cada uno de los modelos entrenados, estos hiperparámetros son los que mejores resultados dieron durante el entrenamiento.

2.4 Evaluación

La evaluación de los modelos permite ajustar valores de entrenamiento y entender los resultados obtenidos. Para la evaluación de los dos modelos CNN-LSTM se utilizan métricas comunes en modelos de redes neuronales artificiales, estas métricas son la exactitud del modelo (accuracy) y la función de perdida (loss).

- Accuracy (Exactitud): El accuracy es una medida de qué tan bien clasifica los datos de prueba el modelo de red neuronal. El accuracy se calcula dividiendo el número de predicciones correctas por el número total de muestras [21].
- Loss (Pérdidas): El loss es una medida de qué tan bien el modelo aprende durante el entrenamiento. La pérdida representa la diferencia entre las predicciones del modelo y los valores reales de los datos de entrenamiento [21].

Estos modelos tienen como objetivo aumentar el valor de accuracy y reducir el valor de loss, por lo tanto, durante la evaluación se verificará dichos valores y se escogerá las mejores opciones. En la Figura 12 se muestra una gráfica del cambio del accuracy durante un entrenamiento del modelo de gestos estáticos con respecto a los epochs aplicados. En esta figura 12 se observan dos funciones, la de color azul representa la exactitud del modelo sobre los datos de entrenamiento y la de color naranja la exactitud en datos de validación.

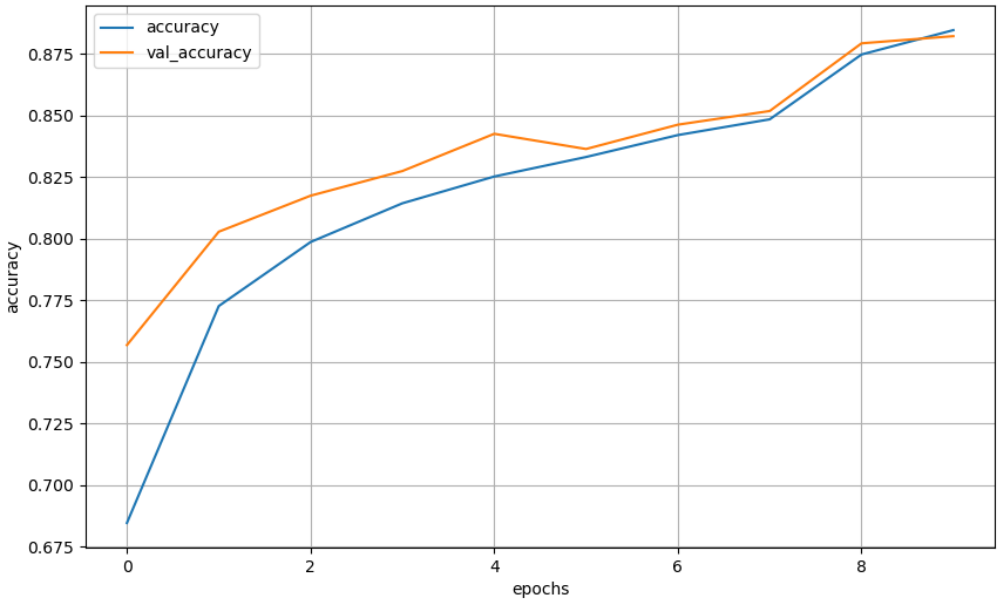


Figura 12 Accuracy durante el entrenamiento del modelo de gestos estáticos

Por otro lado, en la Figura 13 se muestra el cambio del loss durante el entrenamiento del mismo modelo. En la figura hay dos funciones, la azul es la perdida en datos de entrenamiento y la naranja la perdida en datos de validación.

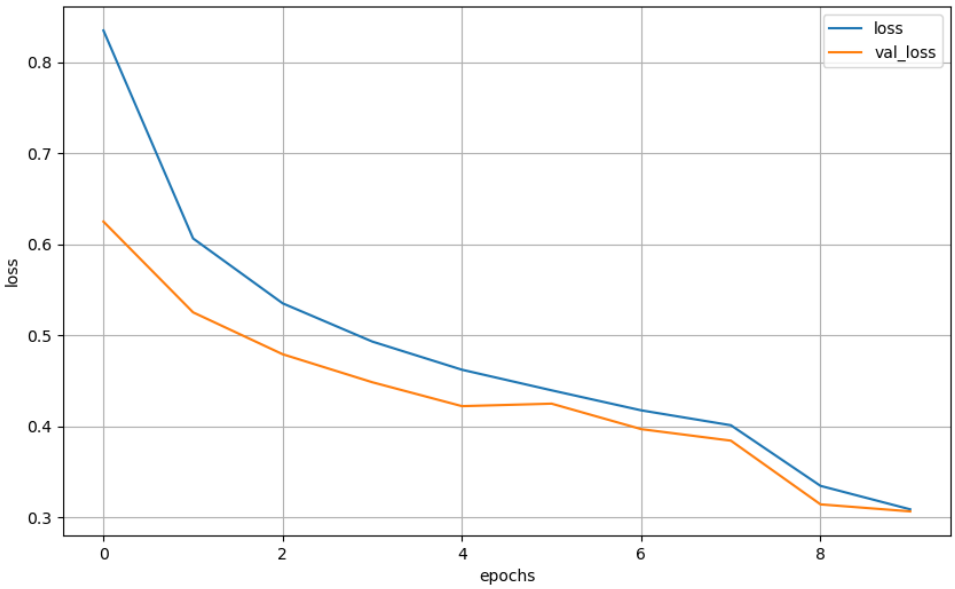


Figura 13 Loss durante el entrenamiento del modelo de gestos estáticos

Los datos de validación son datos que no se utilizan durante el entrenamiento y sirven como una guía para conocer cómo reacciona el modelo a datos desconocidos para el modelo y de esta forma verificar la existencia de overfitting en el modelo.

2.5 Implementación

La fase de implementación de este trabajo consiste en generar un archivo JSON con la clasificación y reconocimiento de los gestos de 42 usuarios que no están etiquetados. Este archivo JSON contiene las predicciones de datos que no se conocen, este archivo es enviado a la aplicación desarrollada en el componente 2 para realizar la evaluación y obtención de resultados. En la Figura 14 se ilustra el proceso de generación del archivo JSON.

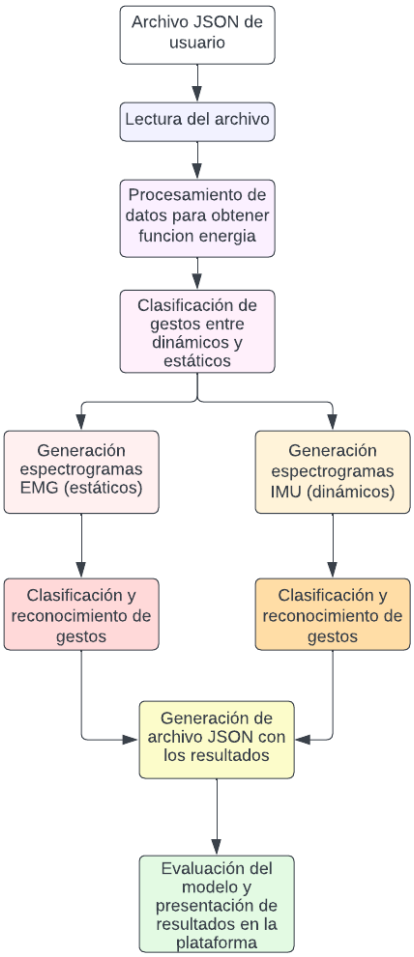


Figura 14 Proceso completo de clasificación y reconocimiento de gestos para datos de prueba

En la fase de implementación se integran los 3 modelos desarrollados, por lo tanto, el programa debe ingresar los datos completos de las señales y dar como resultado un archivo JSON con los gestos clasificados y con su reconocimiento. El resultado que muestra el componente 2 permite conocer la eficacia del modelo con datos totalmente nuevos, ya que durante las predicciones hechas no se conocen las etiquetas reales.

2.6 Mantenimiento

El objetivo de esta fase es analizar los resultados del modelo ya en producción, que en este caso sería después de la evaluación del componente 2, y considerar cambios en la arquitectura del modelo, hiperparámetros o configuraciones.

En esta fase se realizó dos cambios importantes en los modelos planteados, el primer cambio se da en el modelo de clasificación entre gestos dinámicos y estáticos (switch). Este cambio consiste en remplazar la regresión logística por otro algoritmo de machine learning, el algoritmo escogido es KNN. Este cambio no altera la arquitectura del modelo simplemente se remplaza el algoritmo.

A diferencia de la regresión logística, el algoritmo KNN tiene un parámetro modificable para su entrenamiento el cual es k. Este parámetro representa cuántos vecinos se verifican para clasificar un punto específico dentro de su grupo. Por lo tanto, para el entrenamiento del modelo se especifica los valores de k dentro de [1, 2, 3, 4, 5, 10, 20, 30].

El otro cambio consiste en la adición de una capa más Dropout al modelo de clasificación y reconocimiento de gestos estáticos, dado que durante la fase de evaluación del modelo a partir de cierto punto se daba overfitting en el modelo con los datos de entrenamiento. En la Figura 15 se ve el modelo modificado con la capa que se adiciona, la cual sirve para reducir overfitting, por lo tanto, el modelo resultante es idéntico al de gestos dinámicos.

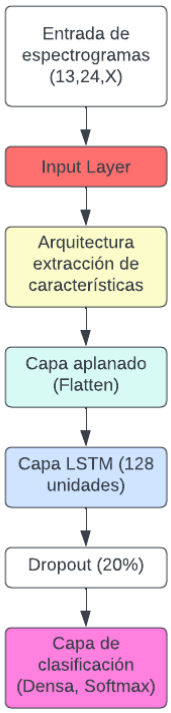


Figura 15 Modelo de clasificación y reconocimiento de gestos estáticos modificado

A diferencia del modelo de gestos dinámicos el porcentaje de dropout de la capa es de 20%, ya que durante la evaluación de este nuevo modelo se pudo comprobar que utilizar un 50% de dropout era perjudicial para modelo y no daba resultados óptimos. En el Anexo III se encuentra el enlace al código desarrollado durante el proyecto.

3 RESULTADOS, CONCLUSIONES Y RECOMENDACIONES

3.1 Resultados

Al terminar el ciclo de la metodología CRISP-ML(Q) y con los mejores resultados obtenidos del entrenamiento de los modelos se realiza el análisis de estos datos. El análisis de resultados se realiza de forma separada para cada modelo y también para los modelos unidos. Para analizar los resultados de cada modelo se utilizan matrices de confusión y para la unión de modelos los resultados obtenidos por la aplicación de evaluación.

Modelo de clasificación KNN (Switch)

En la Figura 16 se muestra la matriz de confusión del modelo KNN de clasificación de gestos dinámicos y estáticos. En esta matriz se muestra los valores que el modelo a clasificado frente a los valores reales.

Como ejemplo se puede ver que en el valor real Dynamic y el valor predicho Dynamic se tiene 685 ejemplos, es decir el modelo predijo 685 valores como dinámicos cuando estos son dinámicos, en cambio cuando el valor real es Static y el valor predicho es Dynamic hay 132 ejemplos, por lo tanto, el modelo clasifico mal 132 ejemplos como dinámicos cuando estos eran estáticos.

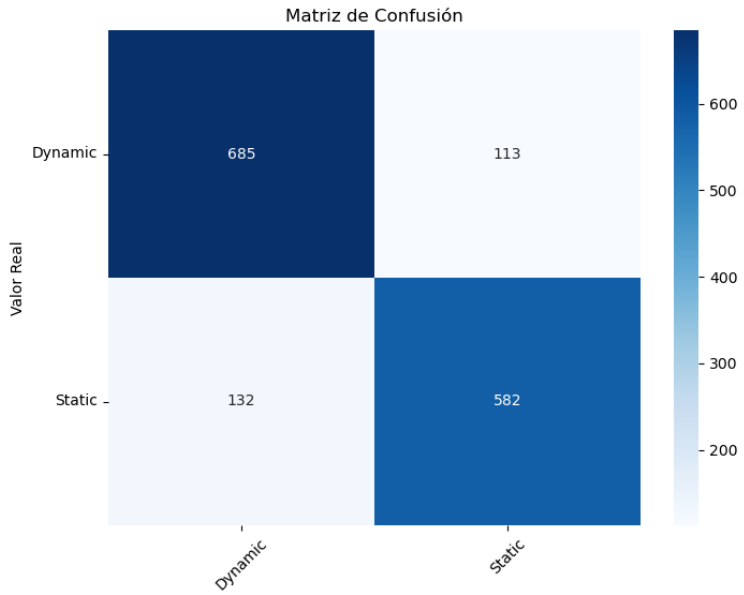


Figura 16 Matriz de confusión del modelo KNN

Con los datos de la matriz de confusión se puede calcular la precisión del modelo en base a la Ecuación 1. Por lo tanto, la exactitud de este modelo es de un 83.8%. El resultado obtenido es muy bueno tomando en cuenta que los datos no están balanceados.

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Ecuación 1. Cálculo de la exactitud (accuracy)

Modelo de clasificación y reconocimiento CNN-LSTM (Estático)

Para el modelo de clasificación y reconocimiento de gestos estáticos se tiene la matriz de confusión de la Figura 17, y usando la Ecuación 1 se obtiene un valor de exactitud de clasificación de 90.59%.

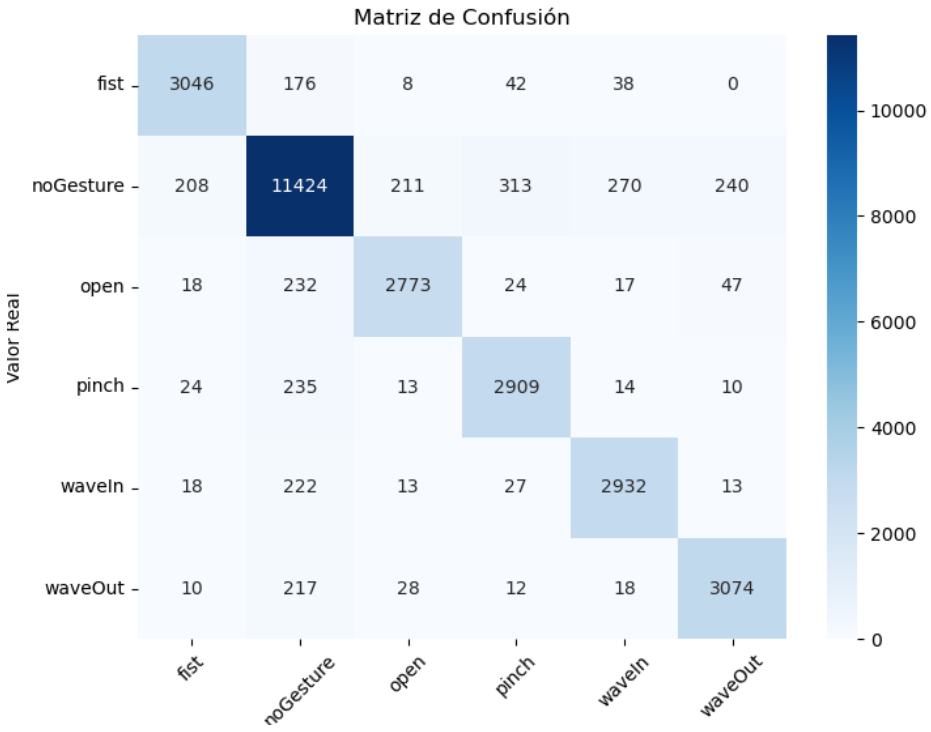


Figura 17 Matriz de confusión del modelo CNN-LSTM gestos estáticos

En la Figura 18 y Figura 19 se observa el accuracy y loss del modelo de gestos estáticos durante el entrenamiento, en estas figuras se muestran los mejores valores de las métricas alcanzados.

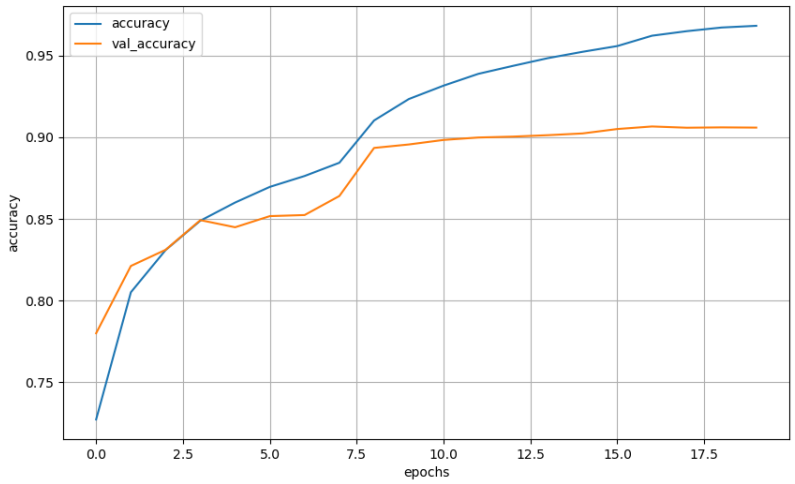


Figura 18 Exactitud durante el entrenamiento del modelo de gestos estáticos

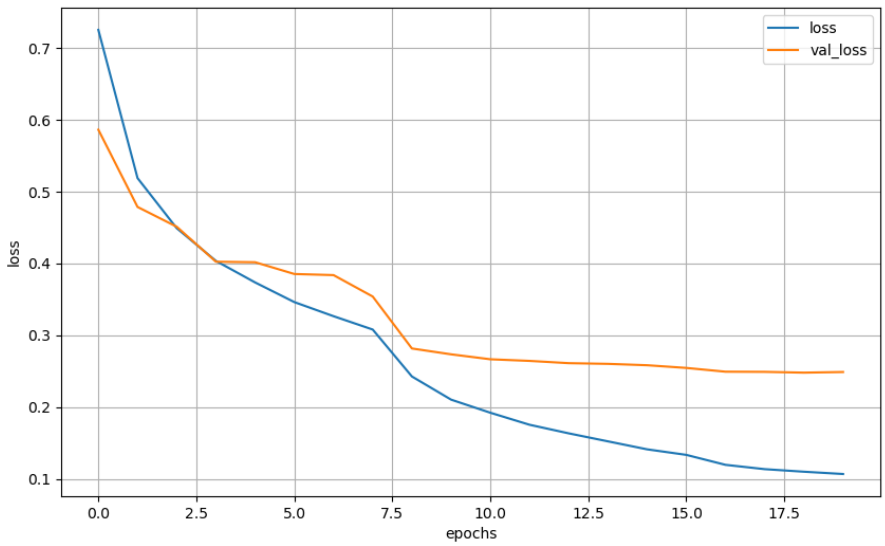


Figura 19 Perdida durante el entrenamiento del modelo de gestos estáticos

El resultado de exactitud alcanzado es alto y de igual forma los valores de perdida son bajos, además, a pesar de los cambios realizados existe un poco de overffiting en el modelo, al acercarse al 90% de exactitud.

Modelo de clasificación y reconocimiento CNN-LSTM (Dinámico)

Para el modelo de clasificación y reconocimiento de gestos dinámicos se tiene la matriz de confusión de la Figura 20. Se obtiene un valor de exactitud de 93.8%, el cual es mayor que en el anterior modelo.

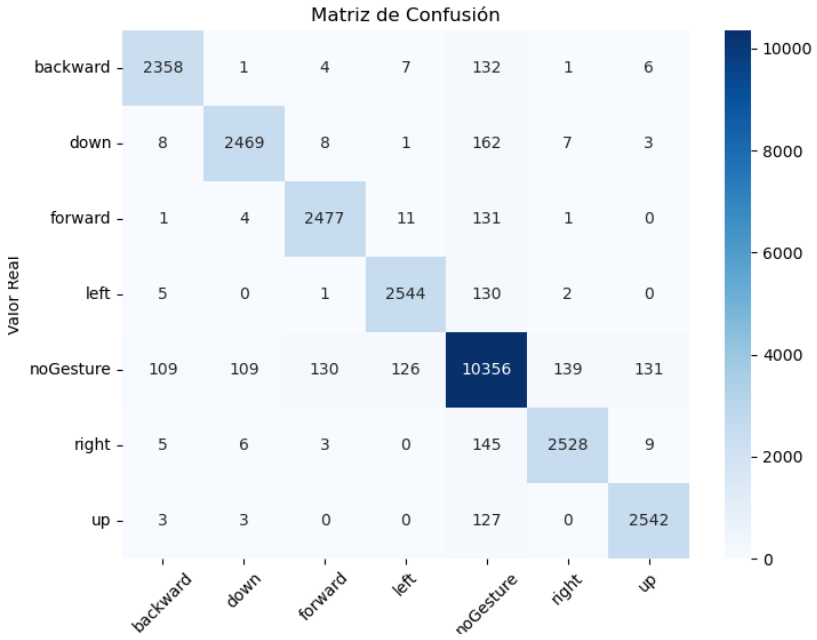


Figura 20 Matriz de confusión del modelo CNN-LSTM gestos dinámicos

En la Figura 21 y Figura 22 se observa el accuracy y loss del modelo de gestos dinámicos durante el entrenamiento. Estas gráficas muestran los mejores valores alcanzados.

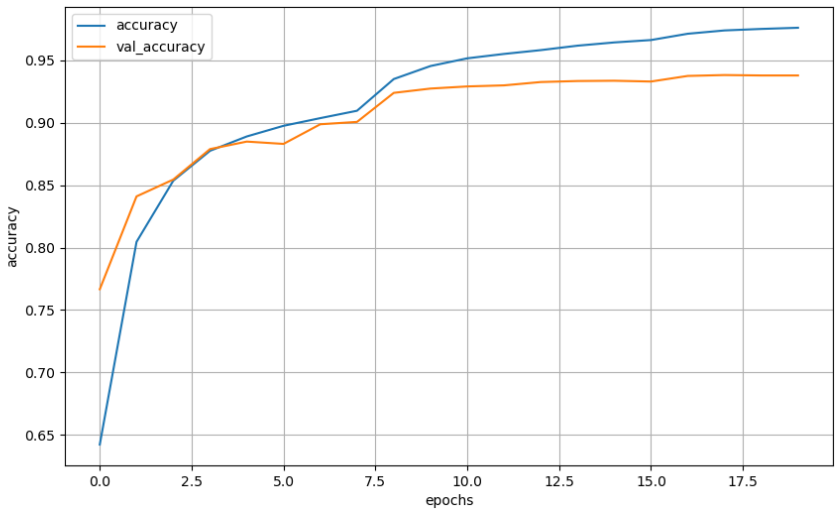


Figura 21 Exactitud durante el entrenamiento del modelo de gestos dinámicos

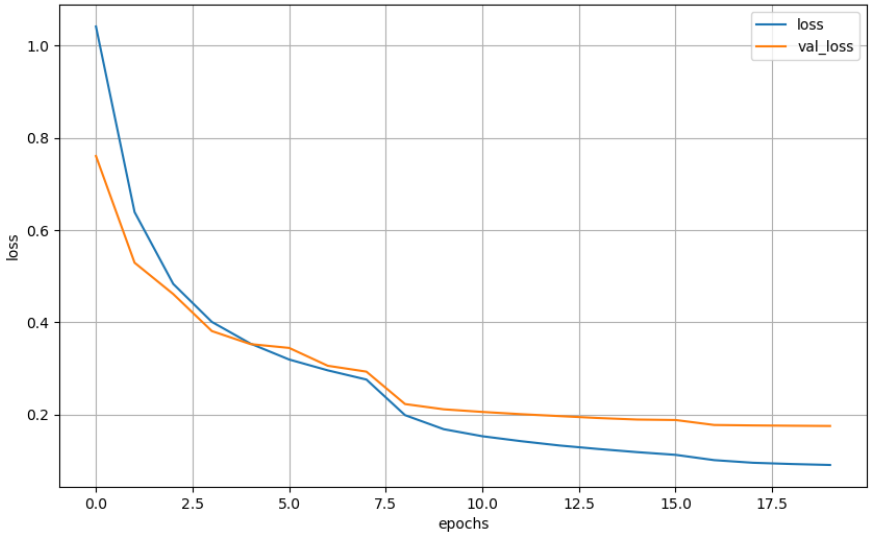


Figura 22 Perdida durante el entrenamiento del modelo de gestos estáticos

En el caso de los gestos dinámicos los resultados de exactitud y perdida obtenidos son un mejores comparados con los resultados obtenidos en el modelo de gestos estáticos. En este caso el modelo no presenta un overfitting tan alto como el del modelo de gestos estáticos.

Clasificación y reconocimiento por la aplicación de evaluación

La aplicación de evaluación desarrollada en el componente 2 muestra una gran cantidad de información a los usuarios que desean evaluar su modelo, en la Figura 23 se muestra un resumen de resultados obtenidos del modelo, principalmente destacan los valores de clasificación del modelo y de exactitud (60.61% de clasificación y un 21.18% de reconocimiento).

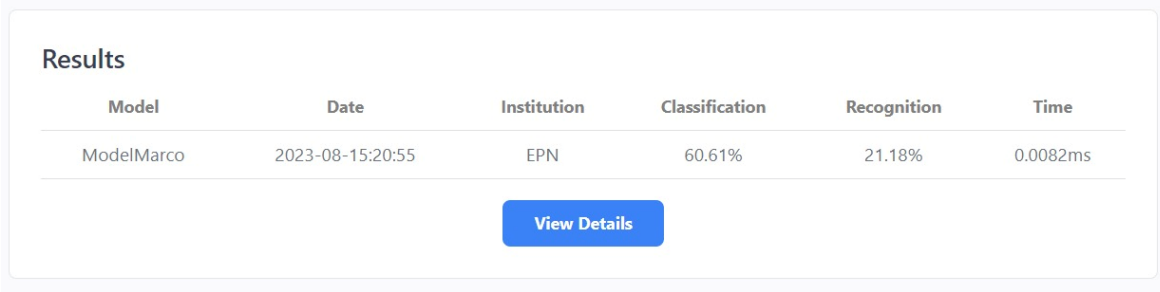


Figura 23 Resumen de resultados de evaluación del modelo

Estos resultados son los promedios de la exactitud de clasificación y reconocimiento de cada usuario, en la Figura 24 se muestran los resultados de la clasificación por usuario, donde la mayoría de los resultados se encuentran entre los valores del 50% y 70% existiendo algunas excepciones que se alejan de la media.

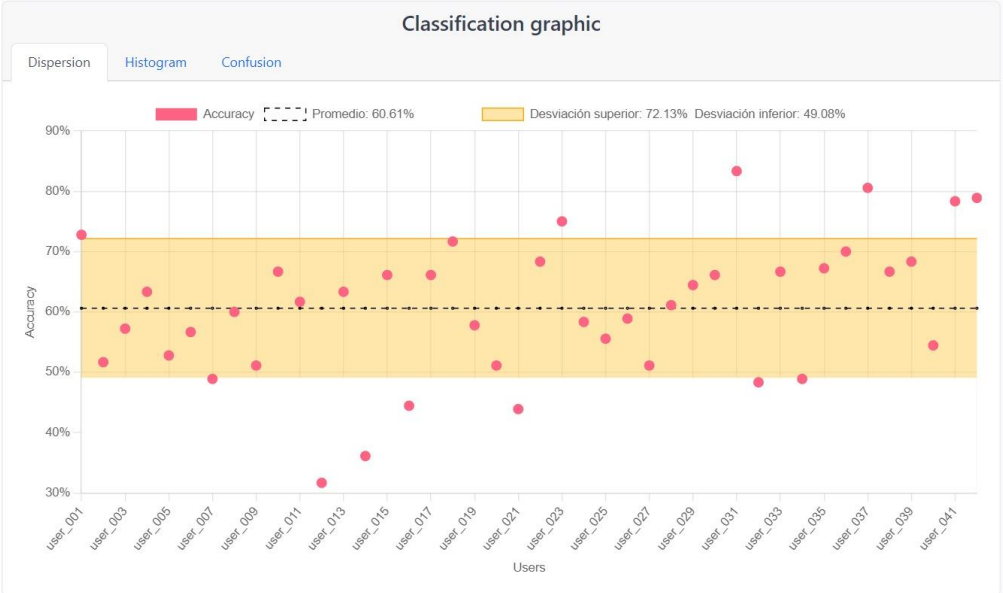


Figura 24 Clasificación del modelo por usuario

En la Figura 25 se muestra los resultados de reconocimiento de los datos con el modelo por usuario en donde los datos están menos dispersos ya que sus valores están más cerca de la media.

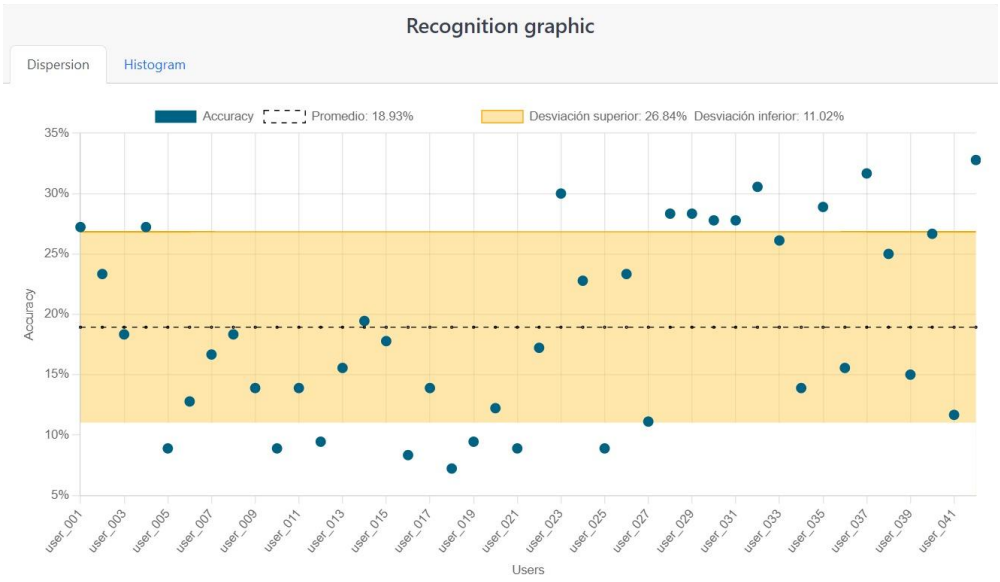
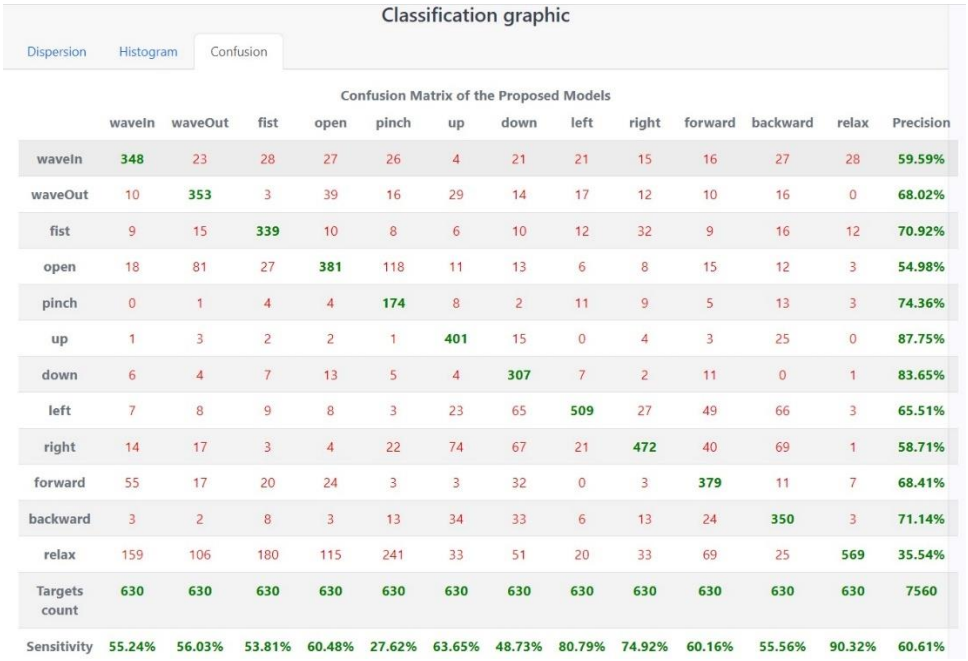


Figura 25 Reconocimiento del modelo por usuario

Finalmente, en la Figura 26 se muestra la matriz de confusión para todos los gestos que clasifica el modelo. Para comprender los resultados de matriz se usará como ejemplo los resultados del gesto open, el resultado en verde (381 ejemplos) representa los gestos predichos como open que si están etiquetados como esa clase. Los valores de la fila open que están en rojo son los gestos etiquetados como open pero que el modelo predijo

como otros gestos. Finalmente, los valores de la columna open en rojo son los gestos que el modelo predijo como open pero en realidad son otros gestos.



	waveIn	waveOut	fist	open	pinch	up	down	left	right	forward	backward	relax	Precision
waveIn	348	23	28	27	26	4	21	21	15	16	27	28	59.59%
waveOut	10	353	3	39	16	29	14	17	12	10	16	0	68.02%
fist	9	15	339	10	8	6	10	12	32	9	16	12	70.92%
open	18	81	27	381	118	11	13	6	8	15	12	3	54.98%
pinch	0	1	4	4	174	8	2	11	9	5	13	3	74.36%
up	1	3	2	2	1	401	15	0	4	3	25	0	87.75%
down	6	4	7	13	5	4	307	7	2	11	0	1	83.65%
left	7	8	9	8	3	23	65	509	27	49	66	3	65.51%
right	14	17	3	4	22	74	67	21	472	40	69	1	58.71%
forward	55	17	20	24	3	3	32	0	3	379	11	7	68.41%
backward	3	2	8	3	13	34	33	6	13	24	350	3	71.14%
relax	159	106	180	115	241	33	51	20	33	69	25	569	35.54%
Targets count	630	630	630	630	630	630	630	630	630	630	630	630	7560
Sensitivity	55.24%	56.03%	53.81%	60.48%	27.62%	63.65%	48.73%	80.79%	74.92%	60.16%	55.56%	90.32%	60.61%

Figura 26 Matriz de confusión del modelo para todos los gestos

3.2 Conclusiones

En este trabajo se ha realizado una revisión exhaustiva del estado actual en el campo de las arquitecturas de redes neuronales aplicadas al reconocimiento de gestos de manos mediante el uso de señales electromiográficas (EMG) y unidades de medición inercial (IMU). Se ha observado que las arquitecturas neuronales, particularmente las redes convolucionales y LSTM, han demostrado un alto grado de eficacia en la extracción y representación de patrones complejos presentes en señales EMG e IMU.

La adopción de archivos JSON con una estructura bien definida ha simplificado significativamente la lectura de los datos, brindando una mayor accesibilidad a cualquier usuario. Este formato de archivo, ampliamente utilizado en diversas aplicaciones del mundo real, ha facilitado tanto el acceso como la manipulación de los datos para futuros análisis.

La metodología CRISP-ML(Q) empleada en el desarrollo de los modelos de inteligencia artificial ha demostrado ser altamente efectiva al permitir un proceso iterativo de mejora continua en los resultados obtenidos. Este enfoque, inspirado en las metodologías ágiles del desarrollo de software, ha acelerado el proceso de obtención de modelos funcionales,

permitiendo la implementación de técnicas específicas para abordar desafíos y mejorar la comprensión de problemas específicos en el contexto del modelo propuesto.

En relación al modelo de reconocimiento y clasificación de gestos de la mano utilizando señales electromiográficas (EMG) y unidades de medición inercial (IMU), los resultados obtenidos en los datos de validación muestran una exactitud de clasificación del 60.61% y una exactitud de reconocimiento del 21.18%.

3.3 Recomendaciones

Después de llevar a cabo un análisis exhaustivo de los resultados obtenidos por el modelo propuesto, resulta de vital importancia optimizar el rendimiento del proceso de clasificación. Esta mejora podría ser alcanzada mediante la adopción de un algoritmo más sofisticado, tal como una red neuronal feedforward, que podría aportar una mayor capacidad de aprendizaje y discriminación.

En relación al conjunto de datos empleado para el entrenamiento de los modelos, se ha identificado un desequilibrio en una de las clases. En casos de esta naturaleza, se recomienda encarecidamente la utilización de la métrica F1 score para el monitoreo del progreso del entrenamiento de la red, en lugar de depender únicamente de la precisión (accuracy). La elección del F1 score se fundamenta en su sensibilidad a la correcta identificación de casos positivos respecto al total de casos verdaderamente positivos.

La implementación de técnicas de equilibrio de datos, tales como el subsampling u oversampling, entre otras posibilidades, para el conjunto de datos empleado, brindará la oportunidad al modelo de adquirir un aprendizaje más preciso y ajustado para cada clase, evitando dificultades inherentes a una sobre-predicción excesiva hacia una clase en particular.

4 REFERENCIAS BIBLIOGRÁFICAS

[1] J. Suarez and R. R. Murphy, "Hand gesture recognition with depth images: A review," 2012 IEEE RO-MAN: The 21st IEEE International Symposium on Robot and Human Interactive Communication, Paris, France, 2012, pp. 411-417, doi: 10.1109/ROMAN.2012.6343787.

[2] M. E. Benalcazar et al., “Real-time hand gesture recognition using the Myo armband and muscle activity detection,” in 2017 IEEE Second Ecuador Technical Chapters Meeting (ETCM), Oct. 2017, pp. 1–6. doi: 10.1109/ETCM.2017.8247458.

[3] S. Studer et al., “Towards CRISP-ML(Q): A Machine Learning Process Model with Quality Assurance Methodology,” Machine Learning and Knowledge Extraction, vol. 3, no. 2, pp. 392–413, Apr. 2021, doi: 10.3390/make3020020.

[4] L. Gila, A. Malanda, I. Rodríguez Carreño, J. Rodríguez Falces y J. Navallas, "Métodos de procesamiento y análisis de señales electromiográficas," Anales del Sistema Sanitario de Navarra, vol. 32, no. Supl. 3, pp. 27-43, 2009.

[5] J. F. Florez, E. Muñoz y O. H. Paruma, "Aplicaciones de las señales bioeléctricas para el control de interfaces hombre-maquina," sin más información de la revista o conferencia en la que fue publicado, 1970.

[6] A. E. Armas-Álvarez, A. K. López-Castañeda, M. A. Díaz y N. A. Barboza, "Control de modelo de prótesis de mano por señal mioeléctrica," en Memorias del Congreso Nacional de Ingeniería Biomédica, vol. 2, no. 1, pp. 328-331, 2015.

[7] P. E. López Martínez y D. F. Merchán Carrillo, "Evaluación y rediseño del sistema de adquisición de señales EMG para aplicaciones de diagnóstico y rehabilitación de pacientes infantiles con parálisis cerebral", 2022.

[8] C. A. Riaño Ríos y V. E. Quintero Machado, "Control de una mano virtual usando señales electromiográficas".

[9] J. Castellanos-Ruíz, L. M. Montealegre-Mesa, B. D. Martínez-Toro, J. J. Gallo-Serna y O. A. Fuentes, "Uso de sensores inerciales en fisioterapia: Una aproximación a procesos de evaluación del movimiento humano," Universidad y Salud, vol. 23, no. 1, pp. 55-63, 2021.

[10] R. Q. Juan y C. M. Mario, "Redes neuronales artificiales para el procesamiento de imágenes, una revisión de la última década," RIEE&C, Revista de Ingeniería Eléctrica, Electrónica y Computación, vol. 9, no. 1, pp. 7-16, 2011.

[11] P. D. Pusiol, "Redes convolucionales en comprensión de escenas" [Tesis de licenciatura], sin más información sobre la institución, 2014.

[12] D. J. León González, "Deconvolución en audio utilizando modelos basados en Machine Deep Learning," sin más información sobre la revista o conferencia en la que se publicó, 2021.

[13] K. B. Prakash, R. K. Eluri, N. B. Naidu, S. H. Nallamala, P. Mishra y P. Dharani, "Accurate hand gesture recognition using CNN and RNN approaches," International Journal, vol. 9, no. 3, 2020.

[14] L. Guasp Alburquerque, "Aplicación de técnicas de Deep Learning para la extracción de términos en dominios específicos" [Tesis doctoral, ETSI_Informatica], 2020.

[15] C. Arana, "Redes neuronales recurrentes: Análisis de los modelos especializados en datos secuenciales" (No. 797), Serie Documentos de Trabajo.

[16] D. S. Team, “Introducción Al Concepto de LSTM - Aprendizaje Automático,” DATA SCIENCE, https://datascience.eu/es/aprendizaje-automatico/comprension-de-las-redes-de-lstm/

[17] D. S. Copaci, J. Arias, L. Moreno y D. Blanco, "Evaluación y desempeño de los sensores Myo Armband y MindRove," en XLIII Jornadas de Automática, pp. 58-65, Universidade da Coruña. Servizo de Publicacións, 2022.

[18] J. M. Muñoz Oña, "Diseño y aplicación de un modelo de reconocimiento de 11 gestos de la mano usando señales EMG y Deep Learning: diseño de un modelo de reconocimiento de 11 gestos de la mano que funcione en tiempo real" [Tesis de licenciatura], Quito: EPN, 2022.

[19] L. I. B. López et al., “An Energy-Based Method for Orientation Correction of EMG Bracelet Sensors in Hand Gesture Recognition Systems,” Sensors 2020, Vol. 20, Page 6327, vol. 20, no. 21, p. 6327, Nov. 2020, doi: 10.3390/S20216327.

[20] R. Dawson, “Why is DevOps for Machine Learning so Different? | HackerNoon,” Nov. 11, 2019. https://hackernoon.com/why-is-devops-for-machine-learning-so-different-384z32f1

[21] J. Martinez Heras, “Precision, recall, F1, accuracy en clasificación,” IArtificial.net, https://www.iartificial.net/precision-recall-f1-accuracy-en-clasificacion/#Accuracy_Exactitud (accessed Aug. 7, 2023).

5 ANEXOS

ANEXO I

Enlace de datos usados para el entrenamiento y evaluación del modelo: https://laboratorio-ia.epn.edu.ec/es/recursos/dataset/2020_emg_dataset_120

ANEXO II

Hiperparámetros del modelo de reconocimiento de gestos estáticos

```
gpu_device = "/gpu:1"
max_epochs = 20
mini_batch_size = 32
initial_learn_rate = 0.0005
learn_rate_drop_factor = 0.2
learn_rate_drop_period = 8
gradient_threshold = 1
validation_patience = 5
```

Hiperparámetros del modelo de reconocimiento de gestos dinámicos

```
gpu_device = "/gpu:1"
max_epochs = 20
mini_batch_size = 64
initial_learn_rate = 0.0005
learn_rate_drop_factor = 0.2
learn_rate_drop_period = 8
gradient_threshold = 1
validation_patience = 5
```

ANEXO III

Enlace del código de los programas desarrollados: https://github.com/laboratorioAI/2023_HGR_Model_Python