PAAKAT: Revista de Tecnología y Sociedad
e-ISSN: 2007-3607
Centro Universitario de Guadalajara

Universidad de Guadalajara
México
paakat@cugdl.udg.mx

Año 15, número 28, marzo – agosto 2025

Desarrollo de Software dedicado a la traducción de la Lengua Mexicana de Señas mediante Deep Learning y Machine Learning

Software Development Dedicated to the Translation of the Mexican Sign Language through Deep Learning and Machine Learning

Héctor Caballero Hernández*
https://orcid.org/0000-0002-2790-833X
Universidad Autónoma del Estado de México

Vianney Muñoz Jiménez**
https://orcid.org/0000-0003-2180-6743
Universidad Autónoma del Estado de México

Marco Antonio Ramos Corchado***
https://orcid.org/0000-0003-3982-6988
Universidad Autónoma del Estado de México

[Recibido: 25/09/2024 - Aceptado para su publicación: 13/11/2024]

DOI: http://dx.doi.org/10.32870/Pk.a15n28.897

Resumen

Existen diversas propuestas para abordar la interpretación de las lenguas de señas (LS) a lenguajes escritos y hablados, empleando desarrollos basados en dispositivos electrónicos y en técnicas de visión computacional. Entre los problemas más comunes para la interpretación de las LS se encuentra la falta de conjuntos de datos estandarizados para generar modelos de interpretación de LS y la ausencia de software multiplataforma. Esta investigación presenta el desarrollo de un software para equipos móviles y de sobremesa dedicado a la interpretación de la Lengua de Señas Mexicana (LSM). Adicionalmente, el software tiene la función de detección de objetos que se encuentren en espacios como casa-habitación y oficinas para su traducción a la LSM y a texto en español para facilitar la integración del vocabulario de la LSM, tanto como para personas sordomudas y sin discapacidad auditiva o del habla. Por otro lado, se ha desarrollado un conjunto de datos para la LSM con 42 clases, basado en distintos escenarios de 30 participantes e imágenes provenientes de internet para desarrollar una red neuronal convolucional (RNC) con el algoritmo YOLOv8 y un modelo de reconocimiento de Machine Learning (ML) mediante MediaPipe, el cual pude ser actualizado a través de un servicio en la nube. Los resultados para la RCN dedicada a la LSM muestran 90% de mAP50 y para el modelo ML un accuracy de 0.817. La RNC dedicada a la traducción de objetos al español consta de 79 clases provenientes del conjunto de datos COCO.

Palabras clave: Inteligencia artificial, conjunto de datos, visión computacional, personas sordomudas, discapacidad, LSM.

Abstract

Several proposals are currently addressing the interpretation of sign language (SL) into written and spoken languages, using developments based on electronic devices and computer vision techniques. Among the most common problems for interpreting SL is the lack of standardized data sets to generate SL interpretation models and the absence of multiplatform software. This research presents software development for mobile and desktop devices dedicated to interpreting Mexican Sign Language (MSL). Additionally, the software can detect objects found in spaces such as homes and offices for their translation into MSL and Spanish text to help the integration of MSL vocabulary for both deaf-mute people and people without hearing or speech disabilities. On the other hand, a dataset for the MSL with 42 classes has been developed based on different scenarios of 30 participants and images from the Internet to develop a convolutional neural network (CNN) with the YOLOv8 algorithm, and a Machine Learning (ML) recognition model using MediaPipe, which can be updated through a cloud service. The results for the CNN dedicated to the LSM show a 90% mAP50, and for the ML model the accuracy is 0.817. The CNN dedicated to the translation of objects into Spanish consists of 79 classes from the COCO dataset.

Keywords: *Artificial Intelligence, dataset, computer vision, deaf and mute people, disability, MSL.*

Introducción

Las personas con discapacidad auditiva y del habla al no poder expresarse, empleando palabras, se enfrentan a limitaciones de comunicación con otras personas, lo cual trunca sus posibilidades de acceso a la educación, empleo y otros servicios (incluidos los digitales). En países donde la lengua de señas (LS) no forma parte del programa educativo, como es el caso de México (Villanueva et al., 2023), provoca altos niveles de frustración (Cruz-Aldrete, 2018).

En México existen más de 2.3 millones de personas que tienen una discapacidad auditiva o del habla, las cuales disponen de la Lengua de Señas Mexicana (LSM) para

poder comunicarse, sin embargo, la mayor parte de la población en México desconoce la LSM, lo cual dificulta la comunicación con personas que no presentan alguna discapacidad (Hernández *et al*., 2023).

Con el desarrollo de la electrónica y de la inteligencia artificial (IA) se han generado herramientas basadas en *hardware* y *software* para interpretar las LS (Juárez-Trujillo et al., 2023). *Machine learning* (ML) se destaca por ser un algoritmo que domina la interpretación de las LS y las redes neuronales convolucionales (por sus siglas en inglés CNN) para clasificar las entradas obtenidas de cámaras RGB y sensores de movimiento (Zepeta *et al.*, 2022).

Tanto los modelos de CNN como los de ML manejan parámetros para validar la efectividad del modelo. Algunas métricas de validación (Powers, 2011) son:

Precisión. Mide la calidad del modelo de ML en tareas de clasificación (ecuación 1).

$$Precisión = \frac{TP}{TP + FP} \tag{1}$$

Recall. Proporciona información sobre la cantidad de casos positivos que el modelo de ML es capaz de identificar (ecuación 2).

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

F1. Se encarga de combinar las métricas precisión y recall a un solo valor. Compara el rendimiento combinado de la precisión y la exhaustividad versus distintas soluciones (ecuación 3).

$$F\,1 = \frac{2 \times (Precisión \times Recall)}{Precisión + Recall} \tag{3}$$

Accuracy. Se encarga de registrar el porcentaje de casos que el modelo ha acertado (ecuación 4).

$$Accuracy = \frac{TN + TP}{TP + FP + TN + FN} \tag{4}$$

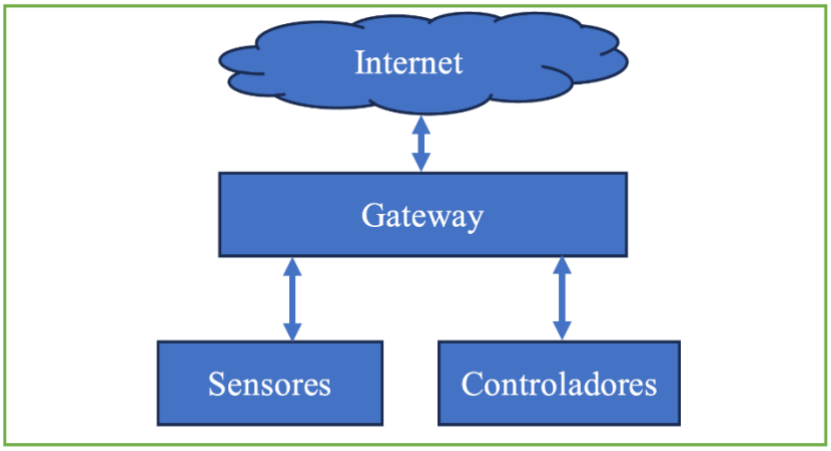
Center loss. La pérdida central incluye un híper-parámetro λ que controla la fuerza de la regularización. Aumentar λ significa aumentar la separación de los centros de clases (Wen et al., 2016).

$$\begin{aligned} L &= L_S + \lambda L_C \\ &= -\sum_{i=1}^{m} log \frac{e^{W_{y_i}^T x_i + b_{y_i}}}{\sum_{j=1}^{m} W_j^T x_i + b_j} + \frac{\lambda}{2} \sum_{i=1}^{m} \| x_i - c_{y_i} \|_2^2 \end{aligned} \tag{5}$$

Donde TP es Verdadero Positivo, TN es Verdadero Negativo, FP es Falso Positivo y FN es Falso Negativo.

Actualmente, entre las tecnologías que destacan por resolver problemas relacionados con la comunicación se encuentra el Internet de las Cosas (IoT, por sus siglas en inglés), el cual permite interconectar elementos físicos a Internet para controlar objetos domésticos, dispositivos médicos, sistemas de conducción, señalización de ciudades, telefonía móvil, entre otros (Mehta *et al*., 2018). En la figura 1 se visualiza la arquitectura de los dispositivos IoT.

Figura 1. Arquitectura de IoT

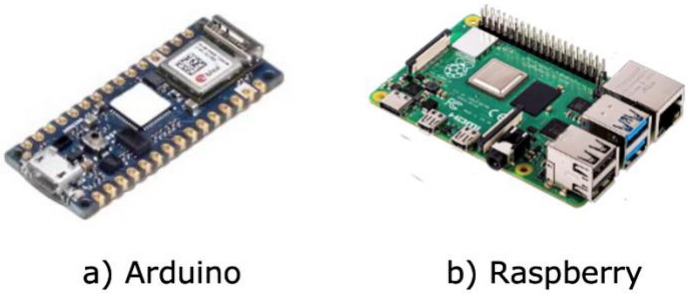


Fuente: Mehta et al. (2018).

La primera capa de la arquitectura IoT se relaciona con los usuarios de Internet. La segunda incluye la red que tiene que ver con los enrutadores para la conexión a Internet. Finalmente, la capa de percepción está relacionada con dispositivos como sensores, actuadores y controladores.

Entre las plataformas de hardware más relevantes relacionadas con el IoT se encuentra Arduino y Raspberry (Novillo-Vicuña *et al.*, 2018) (ver Figura 2), debido a la capacidad de integrarse con diferentes tipos de dispositivos (Quasim *et al.,* 2019) y realizar funciones de lectura de diversas variables, conexión a internet, acceso a base de datos y ejecución de algoritmos avanzados.

Figura 2. Dispositivos IoT más utilizados



a) Arduino b) Raspberry

Fuente: Ferdoush & Li (2014).

La arquitectura de la computación en la nube permite diferentes aplicaciones para la adquisición y manipulación de información sensible en tiempo real, la cual ingresa a los servidores y se utiliza para ser almacenada y procesada. Los servicios en la nube permiten reemplazar parcialmente las aplicaciones de escritorio, ya que hay diversos desarrollos desde procesadores de texto, aplicaciones de vídeo e imágenes y *software* avanzado basado en redes neuronales artificiales (RNA) para reconocer patrones.

La Figura 3 presenta la arquitectura de computación en la nube, que incluye el *software* como servicio (SaaS), la plataforma como servicio (PaaS) y la infraestructura como servicio (IaaS) de los modelos. Todos los modelos de computación en la nube incluyen un riguroso esquema de seguridad, debido a que su violación genera problemas de pérdida de datos, así como mal uso de estos (Palos-Sánchez *et al.*, 2017).

Figura 3. Arquitectura de computación en la nube

Aplicación (Saas)
Software de entorno (Paas)
Software de infraestructura
Recursos computacionales (IaaS) **Almacenamiento de Datos (DaaS)** **Comunicación (CaaS)**
Núcleo
Hardware (Hass)

Fuente: Jones (2008).

En la siguiente sección se presentan investigaciones basadas en la construcción de modelos para interpretar LS desde la perspectiva de la visión artificial y la electrónica.

Estado del arte

Existen investigaciones dedicadas a la interpretación de las LS (se incluye la LSM), usando tecnologías de reconocimiento de patrones basadas en visión computacional e IA. A continuación, se enlistan algunos de los trabajos de mayor relevancia:

- Mejía-Pérez *et al.* (2022) presentan un sistema de reconocimiento automático de LS basado en múltiples gestos, capturando movimientos con una cámara de profundidad (OAK-D) para obtener las coordenadas 3D y clasificarlos mediante redes neuronales recurrentes con una precisión de 97% en evaluaciones sin ruido y 90% de precisión en la detección de los movimientos con ruido.
- Sosa-Jiménez *et al.* (2022) presentaron un sistema traductor bidireccional de LSM para una consulta médica general, empleando un sensor MS Kinect para obtener las trayectorias de las señales e imágenes. Posteriormente, los datos adquiridos son procesados con modelos de ocultos de Markov (HMM). En los experimentos reconocieron 82 signos, obteniendo puntajes de precisión de 99% y F1 de 88%.
- Varela-Santos *et al.* (2021) construyeron un dispositivo que permite identificar 20 señas de la LSM con una cámara integrada a una computadora portátil usando guantes que incluyen marcas de apoyo al generar las señas, obteniendo una precisión de 88 %.
- Sánchez *et al.* (2023) presentaron el diseño de la plataforma Huitzilin, la cual permite al usuario capturar señas en LSM mediante una cámara RGB-3D e indicar la precisión de sus movimientos. Gortarez-Pelayo *et al.* (2023) introducen la plataforma DAKTILOS, la cual trabaja mediante *MediaPipe* (Lugaresi *et al.*, 2019), generando una mano en 3D para representar 27 letras del alfabeto en una plataforma WEB para la enseñanza de la LSM a personas sordomudas.
- Trujillo-Romero *et al.* (2023) obtuvieron 90,000 muestras de 570 palabras y 30 frases interpretadas por 150 personas para crear un conjunto de datos de la LSM. De esta forma, mediante RNA y un sensor Kinect, muestran un modelo con una precisión de 98.62%. González-Rodríguez *et al.* (2024) presentan un sistema para establecer comunicación bidireccional mediante el ingreso de señas o texto; emplearon el modelo BRNN que logra una precisión de 98.8 %.
- Ríos-Figueroa *et al.* (2022) presentan una propuesta capaz de identificar 21 letras mediante un sensor 3D. La precisión obtenida es de 100% en condiciones de laboratorio; el algoritmo principal es *MediaPipe*. Morfín-Chávez *et al.* (2023) presentan la implementación de ML mediante *MediaPipe* con 21 puntos claves de las manos; lograron clasificar 21 letras con una puntuación F1 de 0.98.
- Martínez-Sánchez *et al.* (2023) crearon un conjunto de datos compuesto por un léxico de 100 gestos y 5000 vídeos con diferentes elementos gramaticales con una precisión de 99%. Otras investigaciones como la de Basnin *et al*.

(2021) utilizan los modelos de clasificación CNN y *Long Short Term Memory* (LSTM), basados en un conjunto de datos de 13,400 imágenes, obteniendo una precisión de 88.5% para el lenguaje de señas bengalí (BanSL).

- Breland *et al*. (2021) en su investigación muestran un sistema de reconocimiento basado en un conjunto de datos de 3,200 imágenes térmicas de gestos digitales. El entrenamiento del sistema se basa en redes neuronales profundas con una precisión de 99.5%; el modelo fue implementado en un Raspberry Pi. Sincan & Keles (2020) generaron un conjunto de datos de 38,336 vídeos correspondientes a 226 signos de la lengua turca (LST) grabados mediante un Kinect v2, combinando una CNN y LSTM con 95% de precisión.

Es preponderante el papel de la IA para la interpretación de diferentes LS en sistemas que combinan visión computacional, así como sensores para determinar las coordenadas que está efectuando el signante. Las RNA, CNN y ML se destacan ampliamente para ser implementadas en diferentes equipos de cómputo y dispositivos IoT.

Es importante señalar que la construcción del conjunto de datos estandarizados ayuda a desarrollar mejores modelos de traducción de LS que permitan integrar a la comunidad sordomuda con la sociedad actual. Una de las limitaciones encontradas en la revisión del estado del arte es que los sistemas de reconocimiento de LS no pueden incorporar nuevas señas a su base de conocimiento de forma automática, por tal motivo en la siguiente sección se aborda una propuesta que permita la interpretación de la LSM, así como la incorporación de un mecanismo que permita actualizar la base del conocimiento.

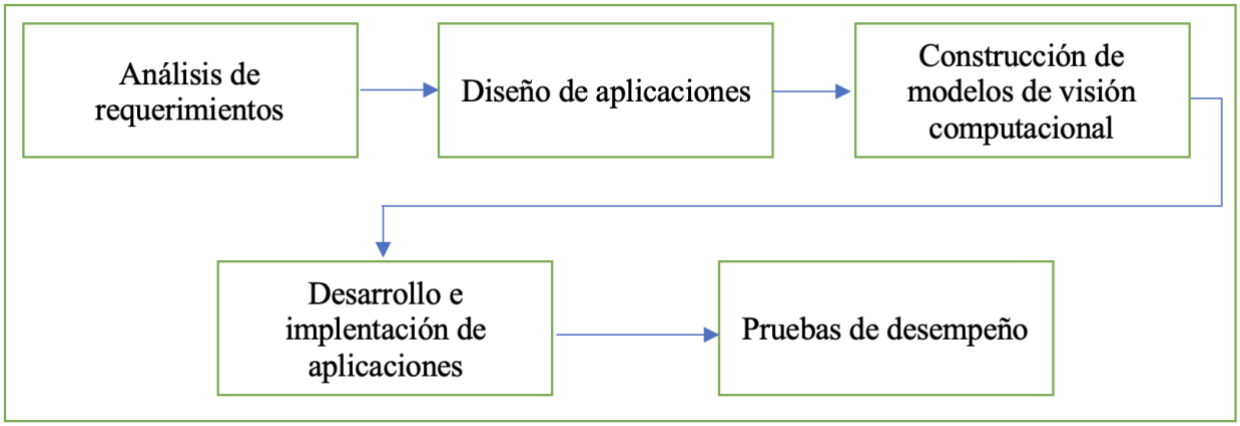
Metodología

En la Figura 4 se presenta la metodología propuesta del desarrollo del software para dispositivos móviles y de sobremesa, que permite reconocer señas de la LSM mediante modelos CNN y ML para realizar traducciones al lenguaje español.

Análisis de requerimientos

En esta etapa se desarrolló un estudio sobre los requisitos fundamentales que debe de cubrir el software, tomando como consideración la capacidad de traducción de las señas LSM, así como la función de presentación de nuevo léxico para los usuarios finales; la capacidad de aprender nuevas señas para la versión de equipos de sobremesa. La Tabla 1 resume los requerimientos por aplicación.

Figura 4. Metodología sugerida para la construcción del traductor de LSM-español



Fuente: elaboración propia.

Tabla 1. Requisitos para la construcción de aplicaciones

Requisitos	Aplicación móvil	Equipo de sobremesa
Traducción de signos LSM al español escrito para letras, dígitos y palabras como casa, papá y hola, entre otras.	SÍ	SÍ
Reconocimiento de objetos físicos y digitales para asociarlos con su interpretación en LSM y español.	SÍ	SÍ
Función de diccionario de consulta de señas.	SÍ	NO
Función de escritura de táctil para el desarrollo de ideas.	SÍ	NO
Capacidad de adquisición de nuevas señas por parte del usuario final.	NO	SÍ

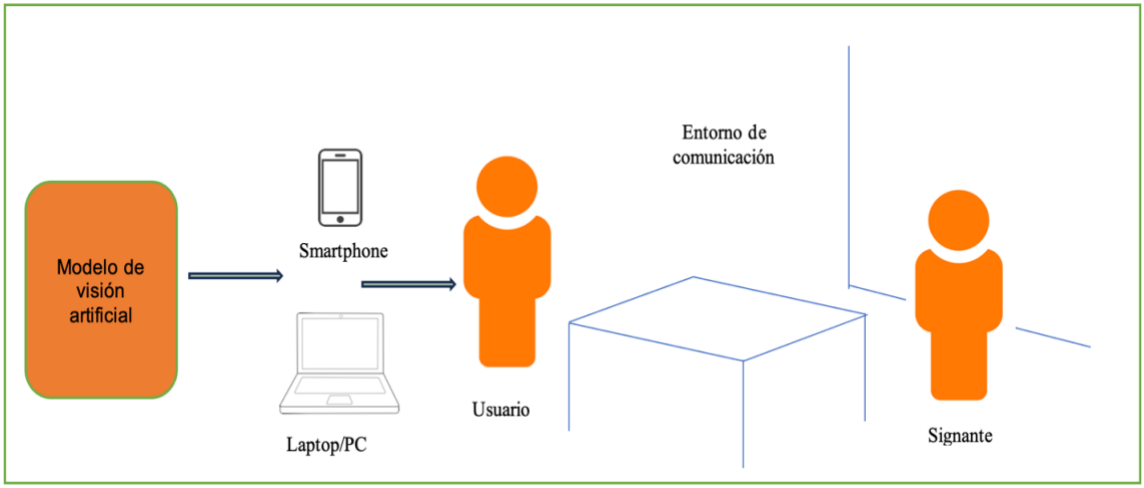
Fuente: elaboración propia.

Cabe señalar que los requisitos ubicados en el antepenúltimo y penúltimo lugar no están disponibles para equipos de sobremesa, debido a que no es ergonómico su funcionamiento para los usuarios como sí lo es al emplearse en dispositivos móviles, mientras que el último requisito no está contemplado para dispositivos móviles. En esta etapa de la investigación, debido a la dificultad que existe al capturar varios fotogramas en la posición fija, se debe de adoptar el usuario.

Diseño de aplicaciones

El software debe de ejecutarse en dispositivos móviles y equipos de sobremesa en entornos donde existan focos de iluminación homogéneos para capturar las imágenes digitales de manera adecuada. En la Figura 5 se presenta el escenario de desenvolvimiento de los dispositivos que emplean los modelos de reconocimiento.

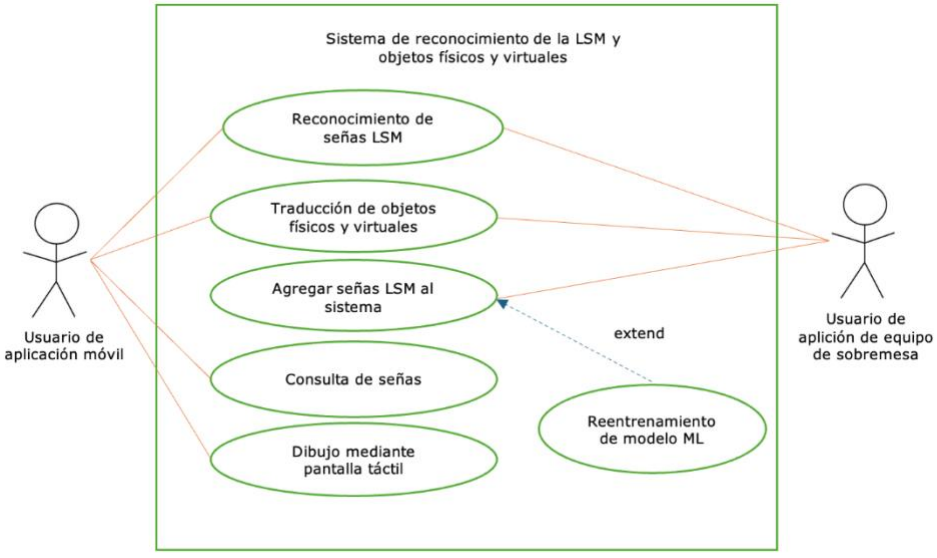
Figura 5. Entorno de desempeño del software y dispositivos



Fuente: elaboración propia.

En la Figura 6 se presenta el diagrama de casos del sistema. Los usuarios de la aplicación móvil pueden acceder a las funciones de reconocimiento de señas LSM, traducción de objetos a LSM y español, consulta de señas LSM mediante un diccionario, así como a la función de dibujo, mientas que los usuarios de equipos de sobremesa pueden ejecutar funciones de reconocimiento de señas LSM, traducción de objetos a LSM y español y agregar nuevas señas mediante la conexión a un servicio en la nube.

Figura 6. Diagrama de casos de uso para el sistema traductor de LSM



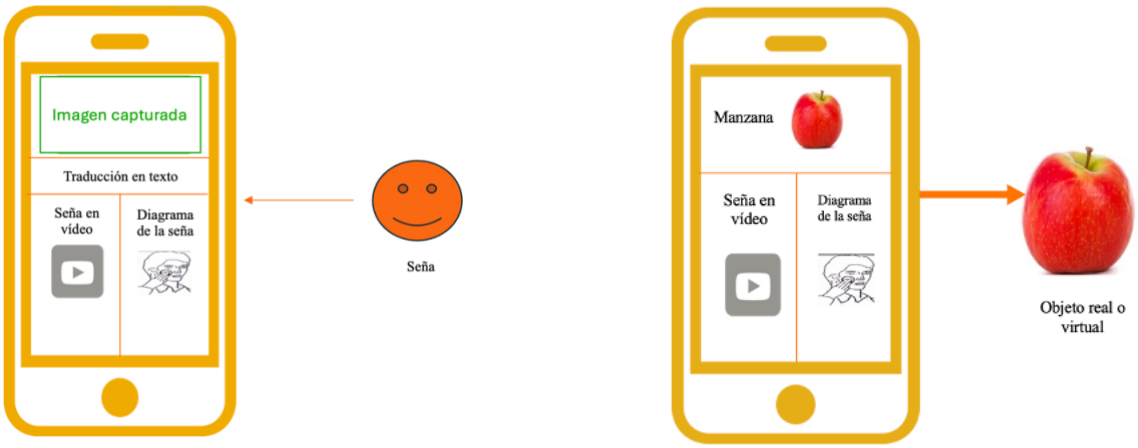
Fuente: elaboración propia.

Para la aplicación móvil se emplean dos modelos de CNN. La primera CNN tiene el objetivo de identificar señas LSM y generar su equivalencia al español escrito, empleando un conjunto de datos de desarrollo propio en la Figura 7: a) mientras que la segunda CNN se ha basado en el conjunto de datos COCO (Lin *et al*., 2014) para identificar objetos físicos o digitales con su correspondiente traducción al español escrito y a LSM mediante imágenes digitales. Figura 7 b) tiene la finalidad de incrementar el léxico de los usuarios.

La aplicación para equipos de sobremesa tiene la función de agregar nuevas señas a su base de conocimiento, empleando un servicio de cómputo en la nube. La figura 8 resume el proceso de incorporación de nuevas señas, el cual consiste en capturar n cantidad de fotogramas del usuario ejecutando la seña. Posteriormente, se comprimen y codifican para ser enviados al servicio de almacenamiento en la nube, estos fotogramas son decodificados y descomprimidos para anexarse al conjunto de datos preexistente, finalizado el proceso.

El conjunto de datos es empleado para entrenar y validar un nuevo modelo ML mediante *MediaPipe* (no requiere etiquetado de imágenes como en una CNN tradicional), donde el nuevo modelo ML generado es obtenido por el usuario a través de la aplicación para reconocer nuevas señas.

Figura 7. Traductor de objetos físicos, digitales y señas de la LSM empleando texto, imágenes y videos

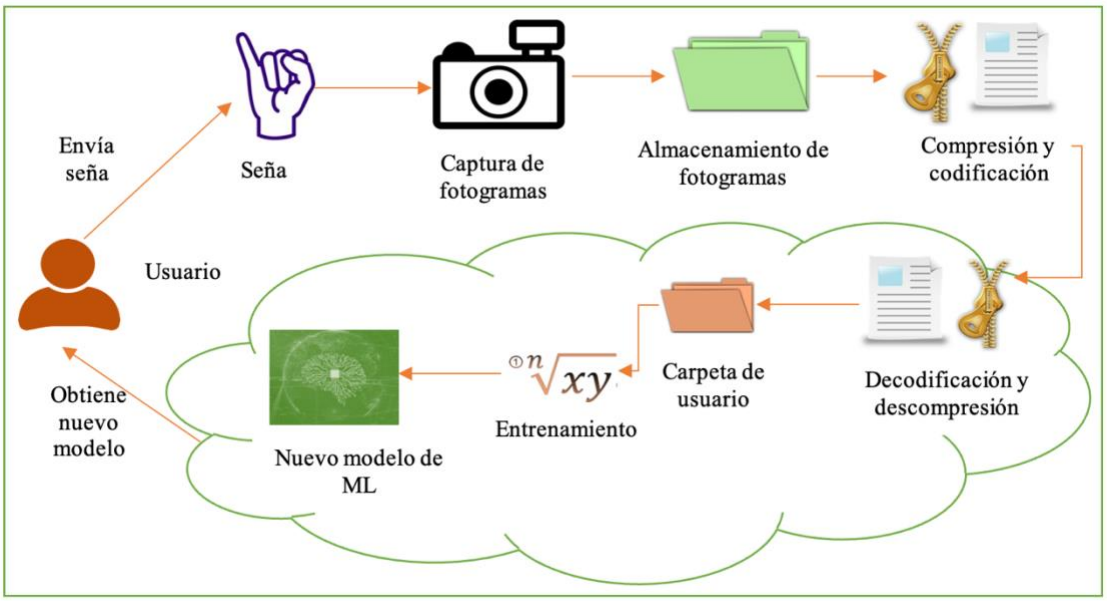


a) Modelo lector de señas

b) Modelo detector de objetos para su traducción a señas LSM

Fuente: elaboración propia.

Figura 8. Proceso de captura de señas y generación del nuevo modelo de ML



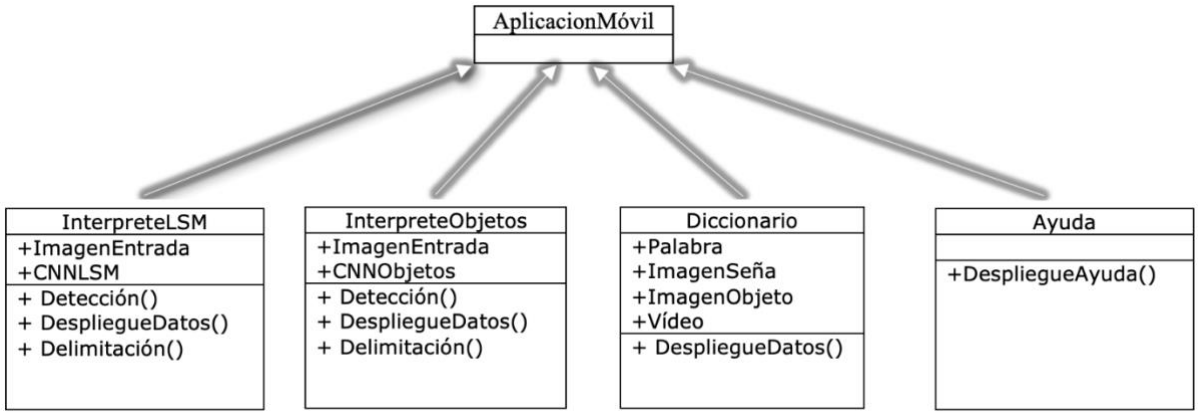
Fuente: elaboración propia.

El diagrama de clases contenido en la figura 9 especifica las relaciones que existen entre las clases de las versiones de *software*, la clase principal es AplicacionMovil, Figura 9 a) de esta heredan las clases InterpreteLSM, InterpreteObjetos, Diccionario y Ayuda, mientras que de la clase AplicacionSobremesa; figura 9 b) heredan las clases InterpreteLSM, InterpreteObjetos, CapturadorFotogramas, EntrenadorSeñas y ActualizadorModelo.

Los métodos polimórficos como Detección() se encargan de realizar la identificación de señas LSM y de objetos empleado los datos contenidos en los modelos de reconocimiento denominados como CNNLSM, CNNObjetos y MLObjetos. Por otra parte, DespliegueDatos() se encarga de realizar el etiquetado de las imágenes capturadas y colocar su representación en palabras, imágenes de cómo realizar la seña LSM y desplegarla en pantalla. La clase EntrenadorSeñas está disponible para el servicio en la nube y se comunica con las clases CapturadorFotogramas y ActulizadorModelo para proveer la actualización del modelo ML.

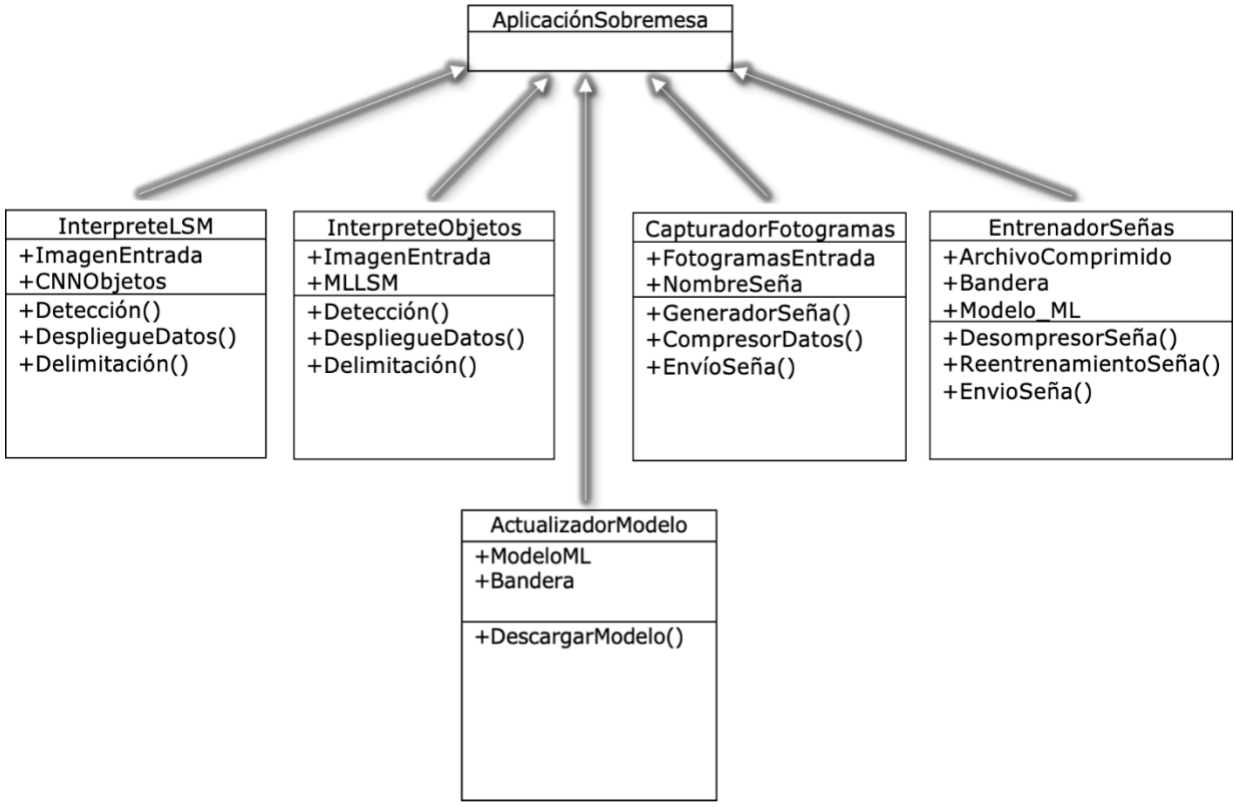
En el Cuadro 1 se presentan las funciones básicas del software de sobremesa disponibles para el usuario de forma directa. El sistema permite el procesamiento de las señas que ha recibido del equipo de sobremesa. En el Cuadro 2 se muestran las fases de entrenamiento de una nueva seña.

Figura 9. Diagrama de clases del software



a) Diagrama de clases para aplicación móvil

b) Diagrama de clases para aplicación de equipo de sobremesa



Fuente: elaboración propia.

Cuadro 1. Funciones del sistema de computación en la nube

1.- Captura del nombre de la seña	Se escribe en texto el nombre de la seña a capturar.
2.- Captura de fotogramas	El sistema captura la seña mediante fotogramas, posteriormente se codifica y luego se envía a la carpeta de usuario.
3.- Detección de señas	El sistema puede detectar una seña mediante la ejecución del modelo ML para posteriormente indicar en texto el nombre de la seña reconocida.
4.- Cargar modelo nuevo	Cuando el usuario genera nuevas señas para el sistema, la aplicación de escritorio tiene la conexión con el módulo de almacenamiento en la nube para cargar el nuevo modelo disponible. De esta forma se descarga el nuevo modelo de ML y se elimina el anterior.

Fuente: elaboración propia.

Cuadro 2. Entrenamiento de nuevo modelo del módulo 2

1.- Carga de módulos de reconocimiento	Las librerías de ML son cargadas al sistema para el entrenamiento del modelo.
2.- Comprobación de directorios base	Los directorios base como el conjunto de datos inicial y señas agregadas son conformadas en un solo directorio.
3.- Incorporación de nuevas señas	Analiza los nuevos datos de entrada como archivos de texto UFT-8. Los datos se decodifican y descomprimen para convertirlos en carpetas que contienen los fotogramas.
4.- Entrenamiento	Las capetas recién agregadas pasan al directorio que contiene las demás señas para ser leídas por el algoritmo de MediaPipe y se generar un nuevo modelo de reconocimiento de LSM.

Fuente: elaboración propia.

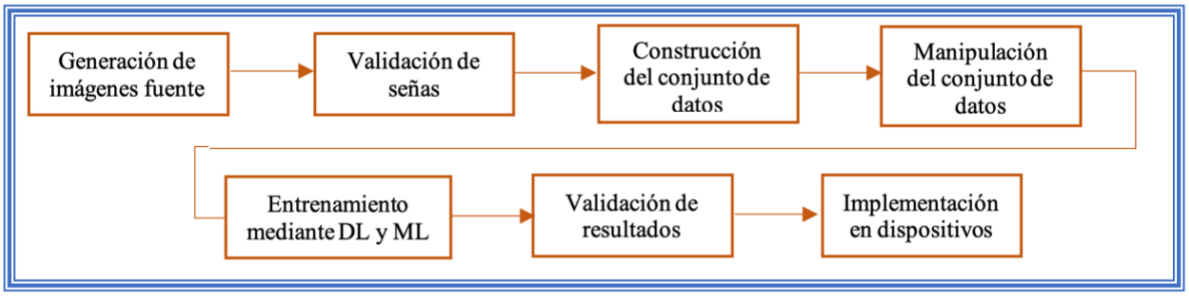
Construcción de modelos de visión computacional

En esta etapa se propone la generación del conjunto de datos dedicado a la detección de la LSM, así como el reconocimiento de objeto físicos y digitales (ver Figura 10).

- Generación de imágenes fuente. Se extraen 420 vídeos provenientes de internet y de 30 participantes que reproduzcan las señas LSM del alfabeto en español, dígitos y palabras como: casa, papá y hola, entre otras. Las señas se graban con cámaras de vídeo convencionales para crear archivos de video MP4. Paralelamente, se obtiene el conjunto de datos COCO (sin mezclarse con los videos iniciales), debido a estas servirán para el modelo de reconocimiento de objetos.

- Validación de señas. Las señas obtenidas se analizan de forma visual para corroborar su validez. Aquellas señas que incumplan la norma de la LSM se descartan.
- Construcción del conjunto de datos. Los vídeos validados se seleccionan para extraer 40 imágenes en formato PNG por cada clase; en total se conforman 42 clases en un conjunto de datos.
- Manipulación del conjunto de datos. Las imágenes contenidas en el conjunto de datos son etiquetadas y redimensionadas a 640 x 640 píxeles y multiplicadas por un factor de 6 para aplicar efectos de rotación; el resultado permite generar un modelo de CNN. Del conjunto de datos original se genera un segundo conjunto de datos donde solo se captura la zona donde se genera la seña con la mano; este será la base para el desarrollo de modelos de ML.
- Entrenamiento mediante DL y ML. Las imágenes contenidas en el conjunto de datos LSM y COCO son procesados mediante el algoritmo YOLOv8 (Redmon *et al.*, 2016) para generar los modelos CNN de reconocimiento de LSM y objetos. El modelo de ML se genera mediante el algoritmo de *MediaPipe*. Para la CNN detectora de objetos se entrena con 1000 épocas para aproximarse a los máximos valores que propone el algoritmo YOLOv8 *small*.
- Validación de resultados. Se verifica que los resultados del modelo de CNN para LSM superen los 0.8 puntos de mAP50. Para el modelo de ML obtener un *accuracy* de 0.8 puntos, de lo contrario se repite el proceso.
- Implementación en dispositivos. Los dos modelos resultantes de las CNN se adaptan para ejecutarse en plataformas móviles, mientras que solo la CNN identificadora de objetos y el modelo ML se adaptan en equipos de sobremesa.

Figura 10. Flujo de trabajo para la construcción de modelos CNN y ML dedicados a la traducción LSM-español y reconocimiento de objetos

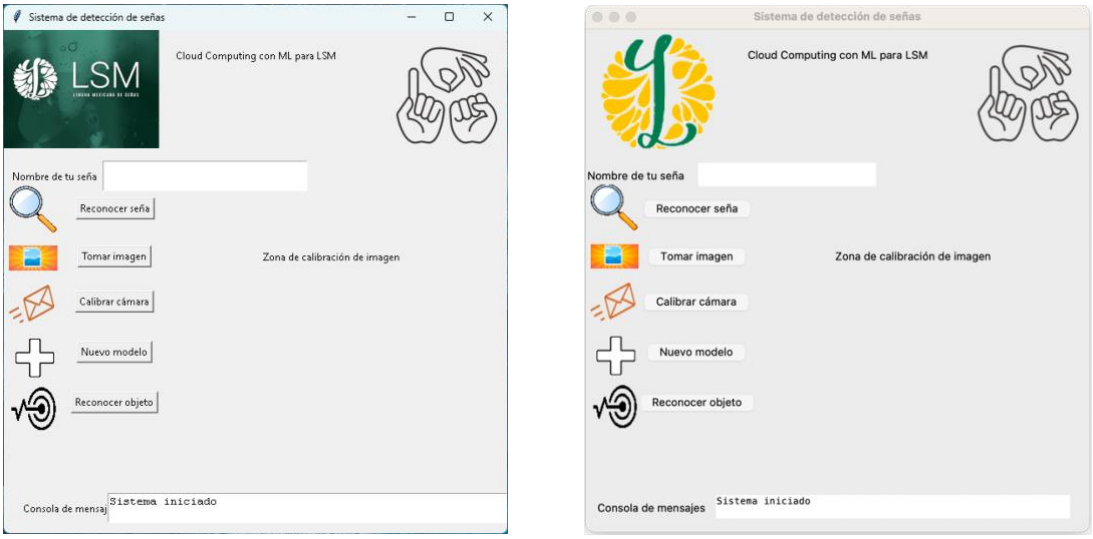


Fuente: elaboración propia.

Desarrollo e implementación de aplicaciones

La Figura 11 presenta el *software* codificado en Python 3.8 para la aplicación de sobremesa consta de siete componentes básicos descritos en el Cuadro 3. La Figura 11 a) presenta la versión para sistemas *Windows* y la Figura 11 b) presenta la versión para sistemas Mac OS 14.x.

Figura 11. Interfaces del sistema de traducción de LSM basado en ML y computación en la nube



a) Interfaz para Windows b) Interfaz para Mac OS X

Fuente: elaboración propia.

Cuadro 3. Componentes básicos del software para la aplicación de sobremesa

Componente	Descripción
1.- Nombre de tu seña	Obtiene la nueva seña mediante la escritura de su nombre y captura la imagen de la seña al agregar al modelo de reconocimiento.
2.- Tomar imagen	Captura la imagen de una seña LSM para agregarse al *software* en la nube. Una vez capturadas 300 fotogramas se envían al servicio de computación en la nube para entrenar el nuevo modelo de ML.
3.- Reconoce seña	Permite al usuario traducir una seña LSM al español mediante la captura de una imagen y el despliegue de su nombre en la consola de mensajes.
4.- Calibrar cámara	Permite que el usuario compruebe si las imágenes que captura el sistema presentan una postura adecuada.
5.- Nuevo modelo	Descarga el nuevo modelo que se encuentra en el servicio de la nube, eliminando el modelo anterior.
6.- Reconocer objeto	Reconoce objetos físicos o digitales mediante una CNN entrenada por el algoritmo YOLOv8. Se analizan las imágenes capturadas y se

	muestra una imagen con objetos localizados con marcos y etiquetas en español.
7.- Zona de calibración de imagen	Presenta una ventana donde aparece la captura que realiza la cámara de vídeo, así como mensajes visuales para informar al usuario lo que ha pasado en determinados procesos.

Fuente: elaboración propia.

El servicio en la nube no cuenta con una interfaz gráfica disponible, por tanto, en esta sección solo se muestran los algoritmos disponibles para la ejecución de este. El Algoritmo 1 detalla el proceso de captura de nuevas señas para el sistema, mientras que el Algoritmo 2 especifica el entrenamiento del nuevo modelo ML, empleando el servicio en la nube.

Algoritmo 1. Proceso de captura de fotogramas para señas nuevas

1. Inicio
2. Definir el nombre de la seña
3. Tomar un conjunto de 300 imágenes en formato PNG de 350x350 píxeles en formato RGB
4. Almacenar las imágenes digitales en un directorio que lleve por título el nombre de la seña
5. Comprimir la carpeta de señas nuevas mediante LZ77
6. Codificar con base 64 y UTF-8 el archivo comprimido y eliminar el archivo comprimido sin codificación
7. Enviar el archivo codificado al directorio del usuario
8. Cambiar el estado a "nueva seña enviada" en el módulo de cómputo en la nube
9. Fin

Fuente: elaboración propia.

Algoritmo 2. Entrenamiento de nuevo modelo ML

1. Inicio
2. Corroborar el estado de nuevas señas
3. Si se tiene una nueva seña, se procede a analizar las entradas
4. Leer el archivo de texto que contiene la nueva seña
5. Decodificar el archivo de texto con B64 y descomprimir la carpeta generada
6. Mover la nueva carpeta al directorio de señas
7. Reentrenar el modelo de ML con la información actualizada
8. Generar el nuevo modelo de reconocimiento de señas y enviar el estado de disponible
9. Fin

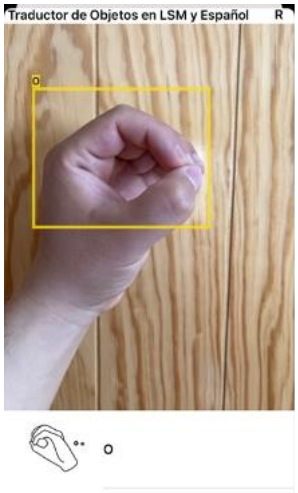
Fuente: elaboración propia.

En la Figura 12 se muestran las interfaces de la aplicación móvil codificada en Swift 5 de la siguiente manera: Figura 12 a), menú principal, Figura 12 b), traducción de objetos a LSM, Figura 12 c), reconocimiento de señas LSM, Figura 12 d), diccionario, y Figura 12 e), dibujo.

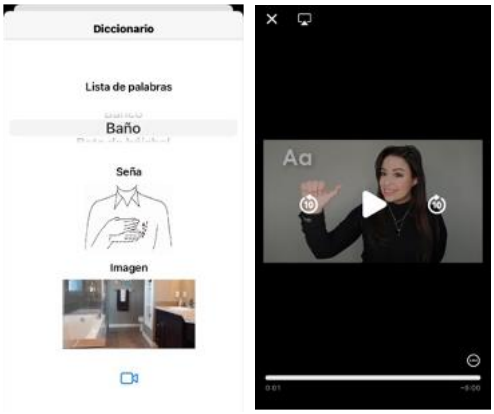
Figura 12. Interfaz de traducción LSM y objetos para dispositivo móvil con sistema iOS

a) Menú principal

b) Detección de objeto con traducción a texto y LSM



c) Interpretación de señas al español



d) Despliegue de palabras con su interpretación en texto, señas y vídeo de la LSM

e) Zona de dibujo para expresión de signos y objetos

Fuente: elaboración propia.

Las funciones principales de la aplicación móvil se presentan a continuación:

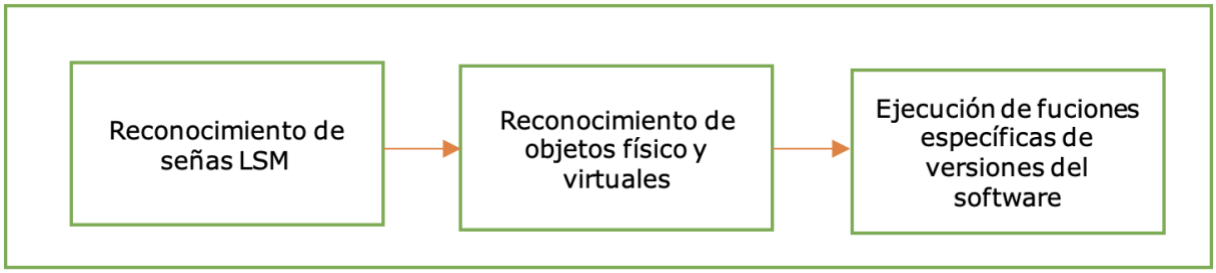
1. Traduce objetos a LSM. Permite traducir objetos físicos o digitales a LSM y al español, empleando texto y diagramas que ejemplifican la seña correspondiente.
2. Reconoce LSM. Reconoce las señas LSM con las cuales se entrenó la CNN y traduce al español mediante etiquetas.

3. Diccionario. Muestra de forma secuencial la interpretación de palabras en LSM con imagen y vídeo, y una imagen que representa la palabra en curso.
4. Dibujar. Permite al usuario escribir o dibujar algún símbolo que le permita comunicarse con otra persona.
5. Ayuda. Muestra instrucciones de cómo utilizar la aplicación.

Pruebas de desempeño

La última etapa de la metodología propuesta consiste en validar la funcionalidad de las aplicaciones móviles y de sobremesa, siguiendo el diagrama de la Figura 13.

Figura 13. Proceso de pruebas para verificar el desempeño de las aplicaciones



Fuente: elaboración propia.

- Reconocimiento de señas LSM. Son ejecutadas en un ambiente donde los focos de iluminación son homogéneos, empleando fondos de superficies mixtas, aplica tanto para dispositivos móviles como de sobremesa. Las señas por reconocer son seleccionadas de forma aleatoria, representando al menos 10% del total de señas que puede reconocer el *software*.
- Reconocimiento de objeto físicos y digitales. Requieren ambientes donde los focos de iluminación sean homogéneos. Los objetos a reconocer se seleccionan aleatoriamente, representando al menos 10% de los elementos disponibles a la versión de COCO de 79 clases.
- Ejecución de funciones específicas de versiones del *software*. Las funciones específicas como diccionario, dibujo (dispositivos móviles) y la agregación de nuevas señas (equipos de sobremesa) son verificadas continuamente para corroborar su funcionamiento.

Pruebas y resultados

1.1 Condiciones iniciales

En esta sección se presentan las pruebas realizadas a los modelos de CNN y ML implementados en dispositivos móviles y de sobremesa para corroborar el desempeño

de la detección de señas de la LSM, objetos y aprendizaje del modelo ML generado por el servicio de computación en la nube. El conjunto de datos destinado al reconocimiento de la LSM contiene 1680 imágenes, 40 imágenes para cada una de las 42 clases y es empleado para generar modelos de CNN y ML.

Para el reconocimiento de objetos se ha generado una CNN versión YOLOv8 *small*, basada en el conjunto de datos COCO con 79 clases. Las condiciones iniciales de los experimentos se observan en la Tabla 2:

Tabla 2. Condiciones de pruebas para corroborar el funcionamiento del software

Tipo de dispositivo	Condición lumínica	Tipo de señas	Espacio de pruebas	Software para la codificación	Modelo de detección de señas y servicios especiales
Teléfono móvil	Foco homogéneo en dispersión lumínica	Aleatorio	Cerrado con superficies mixtas	Swift 5	- 2 CNN basadas en YOLOv8, una entrenada con el conjunto de datos de desarrollo propio para detección de LSM, y la otra para la detección de objetos.
Equipo de sobremesa	Foco homogéneo en dispersión lumínica	Aleatorio	Cerrado con superficies mixtas	Python 3.8.10 *Google-Colab* para el servicio en la nube	- Modelo ML basado en *MediaPipe* para detección del LSM. - CNN basada en YOLOv8, entrenada con el conjunto de datos COCO para detección de objetos. - Enlace de datos mediante Google Cloud y Drive

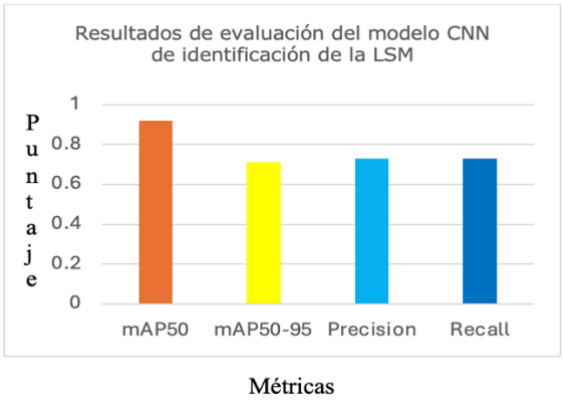
Fuente: elaboración propia.

Las pruebas para el modelo de aprendizaje dinámico se ejecutaron con un modelo ML de 42 clases planteadas inicialmente, mientras que en una segunda prueba se ha cargado un modelo con cinco clases para corroborar la adición de nuevas señas al modelo. Los sujetos de prueba fueron cuatro para emplear el *software* de forma global.

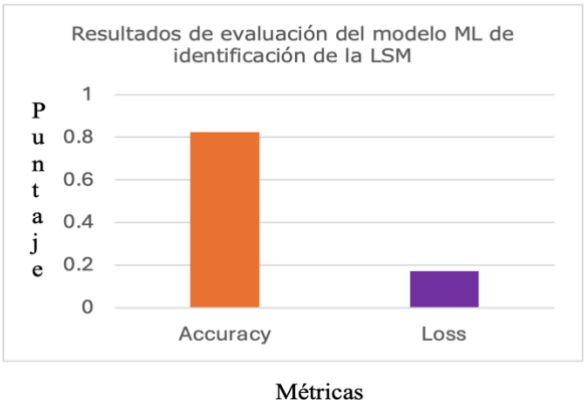
1.2 Resultados de entrenamiento

En la Figura 14 a) se ilustran los resultados obtenidos del entrenamiento de la CNN detectora de la LSM, obteniéndose 0.92 puntos de mAP50, 0.71 puntos de mAP50-95 y finalmente más de 0.73 puntos de *precisión* y *recall*. En el proceso de entrenamiento para el modelo ML basado en *MediaPipe* se obtuvo un *accuracy* de 0.824 y un loss de 0.171, Figura 14 b).

Figura 14. Resultados obtenidos del entrenamiento de los modelos CNN y ML para reconocimiento de la LSM mediante el algoritmo YOLOv8 y MediaPipe



a) Gráfica de resultados de evaluación del modelo CNN de identificación de la LSM



b) Gráfica de resultados de evaluación del modelo ML de identificación de la LSM

Fuente: elaboración propia.

Las características de los equipos empleados para la ejecución de las pruebas se muestran en la Tabla 3. La elección de los celulares inteligentes se debe a que son económicos y de fácil uso, mientras que la elección de los equipos de sobremesa obedece a la cuota de mercado de sus sistemas operativos, Windows y Mac OS.

Tabla 3. Características técnicas de los dispositivos empleados en el proceso de pruebas

Dispositivo	Características técnicas	Condición de la batería
iPhone 8	-Procesador Apple A11 Bionic con 6 núcleos ARMv8-A de 64 bits a 2.39 GHz. -GPU de 3 núcleos. -2 GB de RAM. -Almacenamiento de 64 GB. -Cámara trasera de 12 MP y delantera de 7 MP. -Sistema operativo iOS 16.7.10	76%
iPhone SE 2	-Procesador Apple A13 Bionic con 6 núcleos ARMv8.4-A ISA a 2.65 GHz.	76%

	-GPU de 4 núcleos. -3 GB de RAM. -Almacenamiento de 64 GB. -Cámara trasera de 12 MP y delantera de 7 MP. -Sistema operativo iOS 17.6.1	
Dell Inspirion	-Procesador Intel Pentium Silver N5030 a 3.10 GHz. -8 GB de memoria RAM. -GPU Intel UHD Graphics 605 de 144 núcleos. -Disco SSD 128 GB. -Sistema operativo Windows 11.	100%
MacBook Pro-2017	-Intel Core i5 de doble núcleo y 2.3 GHz. -8 GB de memoria RAM. -SSD de 128 GB. -Mac OS 14.15	94%

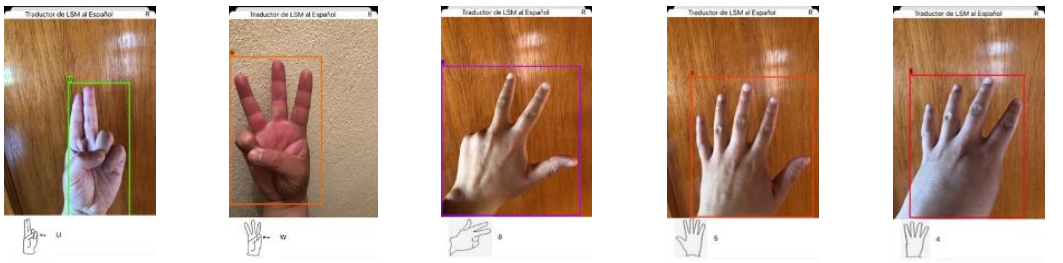
Fuente: elaboración propia.

1.3 Resultados obtenidos del software de reconocimiento de la LSM y objetos físicos y digitales

La Figura 15 contiene los resultados de detección de señas LSM al emplear un iPhone 8. En este caso fue para los incisos a), b), c), d) y e) de la Figura 15. Se detectaron exitosamente las letras u y w, números 8, 5 y 4, respectivamente, mientras que para el iPhone SE 2 corresponden los incisos f), g), h), i) y j) de la Figura 15. Se detectaron correctamente las letras e y u, y los números 5, 1 y 0.

Figura 15. Resultados visuales obtenidos para dispositivos móviles en el reconocimiento de la LSM

iPhone 8



a) Letra U b) Letra W c) Número 8 d) Número 5 e) Número 4

iPhone SE 2

f) Letra E g) Letra u h) Número 5 i) Número 1 j) Número 0

Fuente: elaboración propia

La Figura 16 ilustra la detección de objetos que se encuentran en un entorno casa-habitación por la CNN basada en el conjunto de datos COCO. En la Figura 16 se reconoce en los incisos: a) un microondas; b) una silla; c) una botella de vino; y d) una cama. En la pantalla el usuario observa la traducción al español y su interpretación en LSM como se muestra en los incisos e), f) y g). Cabe resaltar que en los incisos f) y g) son alusivos a una detección de objetos digitales presentados en una pantalla de computadora.

Figura 16. Reconocimiento de objetos a través de la CNN basada en el dataset COCO

a) microondas

b) silla

c) botella

d) cama

e) Detección del objeto silla visto desde la interfaz de la aplicación móvil

f) Detección del objeto digital televisión visto desde la aplicación móvil

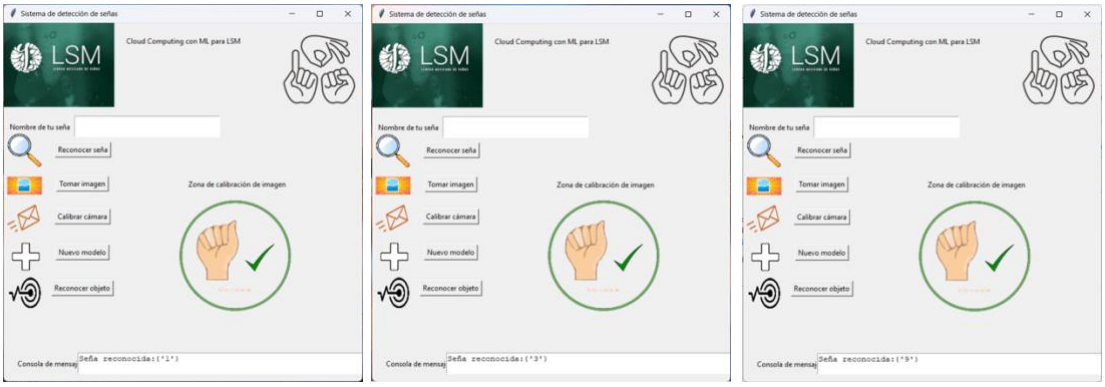
g) Detección del objeto perro en versión digital visto desde la interfaz de la aplicación móvil

Fuente: elaboración propia.

En la Figura 17 a), b) y c) se presenta con éxito el reconocimiento de los dígitos 1, 3, y 9 expresados en LSM, empleando el sistema Windows 11. En la Figura 17 d), e) y f) se presenta con éxito los signos LSM correspondientes a los números 2, 3 y 9, empleando Mac OS 14. En los seis casos se muestra una imagen que indica que la seña se detectó correctamente.

Figura 17. Resultados obtenidos empleando el modelo ML en equipos de sobremesa

Inspirion 3502



a) Número 1 b) Número 3 c) Número 9

MacBook Pro

d) Número 2 e) Número 3 f) Número 4

Fuente: elaboración propia.

En la Figura 18 se pueden observar imágenes del proceso de carga de nuevas señas para la LSM. En este caso se agregaron las letras u y w al modelo que contenía los signos LSM de 1, 6, a, papá y s como se observa en los incisos a) y b) de la Figura 18. En la Figura 18 c) se muestra la carga del nuevo modelo, empleando el botón "Nuevo modelo" y se corrobora el reconocimiento de la seña w recién agregada, Figura 18 d), así como la de la seña a, la cual ya estaba previamente cargada en el modelo Figura 18 e).

En la Figura 19 se muestran imágenes obtenidas durante el reconocimiento de objetos. En la Figura 19 a) y b) se detecta satisfactoriamente una botella y una computadora para el sistema MAC OS-14. En la Figura 15 c) y d) se detecta con éxito el mismo tipo de objetos, pero de distinto tamaño para Windows 11. En este caso solo se muestra la palabra del objeto reconocido.

Figura 18. Adquisición de nuevas señas para el modelo de ML

a) Captura de la seña u

b) Captura de la seña w

c) Carga del nuevo modelo generado en la nube para la aplicación de equipo y sobremesa

d) Reconocimiento de la seña w recién agregada

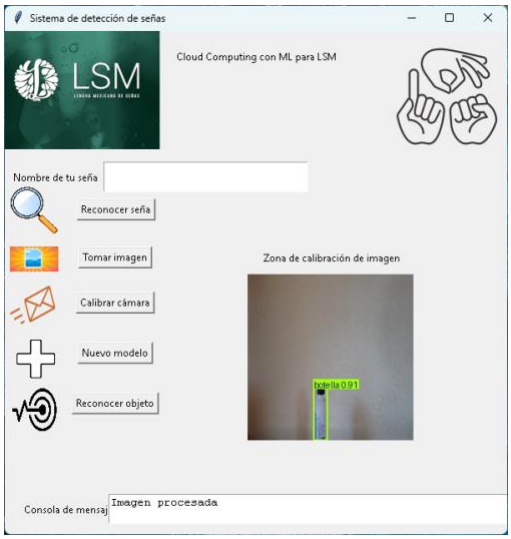
e) Corroboración del reconocimiento de la letra a

Fuente: elaboración propia.

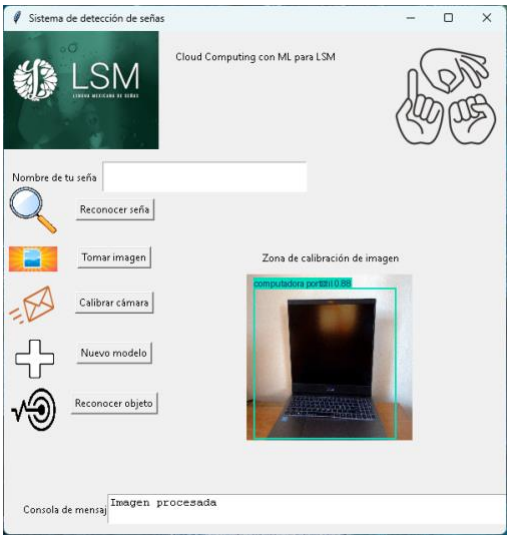
Figura 19. Reconocimiento de objetos en el software de equipos de sobremesa

a) Reconocimiento de una botella para sistema Mac OS 14

b) Reconocimiento de una computadora portátil para sistema Mac OS 14



c) Reconocimiento de una botella para sistema Windows 11



d) Reconocimiento de una computadora portátil para sistema Windows 11

Fuente: elaboración propia.

Discusión de resultados

Tras analizar los resultados se observa que la implementación del *software* con las dos CNN para dispositivos móviles da un desempeño correcto al reconocer las señas de la LSM en fondos homogéneos, donde no presentan efectos notables de reflexión de luz. Así mismo, el reconocimiento de objetos ha generado una traducción correcta a LSM y al español, siempre que se encuentren en el conjunto de datos COCO, permitiendo a los usuarios asociar fácilmente los objetos con el español escrito y la LSM.

Es importante mencionar que tanto la CPU como de la GPU del chip A13 *Bionic* superan el funcionamiento del A11 *Bionic*, permitiendo una identificación de señas LSM y objetos de forma inmediata, mientras que en esta última demora al menos un segundo en realizar este proceso. Sin embargo, el consumo energético en ambos dispositivos se eleva considerablemente al ejecutar las tareas de reconocimiento. Los usuarios de pruebas han manifestado que la aplicación es versátil en los apartados de reconocimiento de señas LSM y objetos, así como la ejecución del diccionario para la asociación de palabras con imágenes y videos. Por otro lado, la función dibujar permite expresar pensamientos mediante dibujos con otras personas que no dominan la LSM, o cuando no saben expresar con señas LSM alguna inquietud.

El *software* desarrollado para reconocer la LSM con aprendizaje dinámico es novedoso, ya que supera las limitaciones de las propuestas clásicas de los sistemas de reconocimiento de LS, donde la base de reconocimiento es estática y depende de las actualizaciones que se realicen de forma centralizada. En esta propuesta, los datos de la base de conocimiento permanecen y no son alterados por el usuario, aunque es importante señalar que la fiabilidad de la respuesta del modelo ML se basa en el despliegue correcto de las señas al capturarlas y de las condiciones lumínicas presentes.

Si bien, el software puede ser adaptado para ejecutarse enteramente en computadores con procesadores Intel, el tiempo de entrenamiento pasaría de minutos a horas al entrenar el modelo de ML con una gran cantidad de clases si no se cuenta con GPUs con una RAM superior a 15 GB, lo cual limitaría ampliamente la adquisición de estos equipos por parte de los usuarios finales. La Tabla 4 resume las ventajas y desventajas del modelo de detección de la LSM y objetos los dispositivos de prueba:

Tabla 4. Ventajas y desventajas de los modelos de detección implementados en los dispositivos

Dispositivo	Ventajas	Desventajas
iPhone 8	- Bajo consumo energético para la detección. - Bajo cansancio visual. - Facilidad de uso en ambientes mixtos.	- Velocidad de detección medianamente fluida. - Visualización en pantalla reducida.
iPhone SE 2	- Respuesta fluida al detectar señas. - Bajo cansancio visual. - Facilidad de uso en ambientes mixtos.	- Consumo poco moderado al detectar señas LSM y objetos. - Visualización de imágenes en pantalla reducida.
Equipos de sobremesa	- Detección en tiempo real. - Facilidad de uso en ambientes cerrados o semiabiertos. - Capacidad de incrementar la base de conocimiento de señas LSM. - Bajo consumo de recursos computacionales en el reentrenamiento del modelo ML.	- Retraso al momento de cargar el modelo de detección de señas. - Consumo energético elevado.

Fuente: elaboración propia.

Al comparar los resultados obtenidos de desempeño del software presentado con investigaciones como la de Mejía-Pérez *et al.* (2022) y Trujillo-Romero *et al.* (2023), la implementación del modelo de CNN en dispositivos móviles permite una mayor interacción con los usuarios al momento de interpretar la LSM al español, su flexibilidad de operación en ambientes con condiciones lumínicas estables, así como la opción en el reconocimiento de objetos para su traducción a LSM y español, el cual no se contempla en otras propuestas.

Sin embargo, los resultados presentados en las propuestas mencionadas son mayores en la precisión del reconocimiento, debido al manejo del conjunto de datos más grande, lo que aporta mayor fiabilidad en entornos con ruido. En comparación con la aplicación DAKTILOS de Gortarez-Pelayo *et al.* (2023) y Huitzilin de Sánchez *et al.* (2023), las aplicaciones móviles permiten no solo reconocer las señas generadas por el usuario en LSM, sino también reconocer objetos físicos o digitales, además no se necesita emplear hardware especializado como el uso de guantes (Varela-Santos *et al*., 2021) para el reconocimiento de la LSM, lo cual permite al usuario mayor grado de libertad.

La combinación del procesamiento de datos en computo en la nube con equipos de sobremesa que empleen ML para el reconocimiento de la LMS permite generar aplicaciones que se ejecuten en equipos de bajo o mediano rendimiento, así como la capacidad de incluir nuevas señas al modelo de reconocimiento sin grandes complicaciones técnicas.

Finalmente, al considerar los resultados de la fase experimental es viable desarrollar aplicaciones para teléfonos inteligentes con sistema operativo Android similares en características técnicas al iPhone 8, mientras que el *software* de ML y computo en la nube puede adaptarse a sistemas Linux por la versatilidad que otorga Python, cubriendo un mayor número de sistemas donde puede ejecutarse.

La tecnología de computación en la nube puede extenderse a dispositivos móviles para acceder a funciones de reconocimiento y traducción de la LSM y de objetos al español con una base de conocimiento dinámica, limitando el consumo energético en estos dispositivos, cuya principal desventaja se encuentra en la capacidad de almacenamiento de carga eléctrica en su batería.

Conclusiones

Este trabajo se ha centrado en desarrollar aplicaciones móviles y de equipos de sobremesa dedicadas a reconocer señas LSM y objetos físicos y digitales, empleando modelos de CNN con el algoritmo YOLOv8 y ML mediante el algoritmo *MediaPipe*. El modelo de CNN dedicado a detectar la LSM ha obtenido resultados superiores a 0.9 puntos de mAP50 y mAP50-95 de 0.71 puntos, lo cual le ha permitido detectar señas LSM en ambientes con fondos mixtos y focos de iluminación homogéneos, permitiendo a los usuarios asociar las señas con palabras en español.

Se ha logrado generar el reconocimiento de objetos para asociarlos con la LSM y el español, mientras que con la combinación de la computación en la nube y el ML en equipos de sobremesa ha sido posible abordar el problema de modelos estáticos de reconocimiento de las LS, lo cual permite la detección de nuevas señas

que no se han contemplado en el modelo original, permitiendo a los usuarios incrementar la base de conocimiento de la aplicación sin una asistencia personalizada. El *software* desarrollado ha sido enfocado a un público que requiera herramientas de fácil acceso para el aprendizaje de la LSM, cuya tecnología no requiera grandes capacidades de cómputo, permitiendo ser de utilidad para personas con o sin discapacidad auditiva o del habla.

Finalmente, los resultados obtenidos en este trabajo dan la pauta para extender esta propuesta a sistemas operativos como Android y Linux, así como un incremento de clases a reconocer mediante modelos de CNN y ML para el contexto de LSM, aunado a la incorporación de las capacidades de la computación en la nube para generar bases de conocimiento dinámicas y reducción de consumo energético.

Referencias

Amorós-Pons, Anna, Comesaña-Comesaña, Patricia e Inna Alexeeva-Alexeev (2022). "Violencia de género en período de pandemia de coronavirus en los países del G-20: Campañas publicitarias en redes sociales", en *Historia y Comunicación Social*, *27*(2), 389-400. https://doi.org/10.5209/hics.84387 14 de agosto de 2023.

Basnin, N., Nahar, L. & Hossain, M. S. (2021). An integrated CNN-LSTM model for Bangla lexical sign language recognition. In Proceedings of International Conference on Trends in Computational and Cognitive Engineering: Proceedings of TCCE 2020, Springer-Singapore, 695-707. https://doi.org/10.1007/978-981-33-4673-4_57.

Breland, D. S., Skriubakken, S. B., Dayal, A., Jha, A., Yalavarthy, P. K. & Cenkeramaddi, L. R. (2021). Deep learning-based sign language digits recognition from thermal images with edge computing system. IEEE Sensors Journal, 21(9), 10445-10453. 10.1109/JSEN.2021.3061608

Cruz-Aldrete, M. (2018). La evaluación del modelo educativo bilingüe para la comunidad sorda en México: un problema sin voz. Voces de la Educación. https://www.revista.vocesdelaeducacion.com.mx/index.php/voces/article/view/91.

Ferdoush, S. & Li, X. (2014). Wireless sensor network system design using Raspberry Pi and Arduino for environmental monitoring applications. Procedia Computer Science, 34, 103-110. https://doi.org/10.1016/j.procs.2014.07.059.

González-Rodríguez, J. R., Córdova-Esparza, D. M., Terven, J. & Romero-González, J. A. (2024). Towards a Bidirectional Mexican Sign Language–Spanish Translation System: A Deep Learning Approach. Technologies, 12(1), 7. https://doi.org/10.3390/technologies12010007.

Gortarez-Pelayo, J. J., Morfín-Chávez, R. F. & López-Nava, I. H. (2023). DAKTILOS: An Interactive Platform for Teaching Mexican Sign Language (LSM). In International Conference on Ubiquitous Computing and Ambient Intelligence, Cham: Springer-Nature-Switzerland, 264-269. https://doi.org/10.1007/978-3-031-48642-5_25.

Hernández, R. R., Jaimes, E. I. G., Mora, V. T., Chau, A. L. & Morán, C. O. G. (2023). Impacto del sistema para la enseñanza y traducción de la lengua de señas mexicana UAEMex en instituciones públicas. Ciencia Latina Revista Científica Multidisciplinar, 7(1), 822-838. https://doi.org/10.37811/cl_rcm.v7i1.4434.

Jones, B. (2008). An EGEE Comparative Study: Grids and Clouds Evolution or revolution. Report comparing EGEE grid to Amazon Web Services, Dated, 11(06).

Juárez-Trujillo, I. A., Zavala de Paz, J. P., Palillero Sandoval, O. & Castillo Velásquez, F. A. (2023). Calibración de cámara multiespectral utilizando redes neuronales convolucionales. Computación y Sistemas, 27(3), 801-810. https://doi.org/10.13053/cys-27-3-4605.

Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D. & Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Springer International Publishing, Proceedings, Part V, 13, 740-755. https://doi.org/10.1007/978-3-319-10602-1_48.

Lugaresi, C., Tang, J., Nash, H., McClanahan, C., Uboweja, E., Hays, M. & Grundmann, M. (2019). Mediapipe: A framework for building perception pipelines. arXiv preprint arXiv:1906.08172. https://doi.org/10.48550/arXiv.1906.08172.

Martínez-Sánchez, V., Villalón-Turrubiates, I., Cervantes-Álvarez, F. & Hernández-Mejía, C. (2023). Exploring a Novel Mexican Sign Language Lexicon Video Dataset. Multimodal Technologies and Interaction, 7(8), 83. https://doi.org/10.3390/mti7080083.

Mehta, R., Sahni, J. & Khanna, K. (2018). Internet of things: Vision, applications and challenges. Procedia computer science, 132, 1263-1269. https://doi.org/10.1016/j.procs.2018.05.042.

Mejía-Pérez, K., Córdova-Esparza, D. M., Terven, J., Herrera-Navarro, A. M., García-Ramírez, T. & Ramírez-Pedraza, A. (2022). Automatic recognition of Mexican Sign Language using a depth camera and recurrent neural networks. Applied Sciences, 12(11), 5523. https://doi.org/10.3390/app12115523.

Morfín-Chávez, R. F., Gortarez-Pelayo, J. J. & López-Nava, I. H. (2023, November). Fingerspelling Recognition in Mexican Sign Language (LSM) Using Machine Learning. In Mexican International Conference on Artificial Intelligence. Cham: Springer-Nature-Switzerland, 110-120. https://doi.org/10.1007/978-3-031-47765-2_9.

Novillo-Vicuña, J., Rojas, D. H., Olivo, B. M., Ríos, J. M. & Villavicencio, O. C. (2018). Arduino y el Internet de las cosas (Vol. 45). 3 ciencias.

Palos-Sánchez, P. R., Arenas-Márquez, F. J. & Aguayo-Camacho, M. (2017). Cloud computing (SaaS) adoption as a strategic technology: Results of an empirical study. Mobile Information Systems, 2017(1), 2536040. https://doi.org/10.1155/2017/2536040.

Powers, David M. W. (2011). "Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation". Journal of Machine Learning Technologies. 2(1): 37–63. https://doi.org/10.48550/arXiv.2010.16061.

Quasim, M. T., Khan, M. A., Abdullah, M., Meraj, M., Singh, S. P. & Johri, P. (2019, December). Internet of things for smart healthcare: a hardware perspective. In 2019 first international conference of intelligent computing and engineering (ICOICE), IEEE, 1-5. 10.1109/ICOICE48418.2019.9035175.

Redmon, J., Divvala, S., Girshick, R. & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition, 779-788. https://doi.org/10.48550/arXiv.1506.02640.

Ríos-Figueroa, H. V., Sánchez-García, A. J., Sosa-Jiménez, C. O. & Solís-González-Cosío, A. L. (2022). Use of Spherical and Cartesian Features for Learning and Recognition of the Static Mexican Sign Language Alphabet. Mathematics, 10(16), 2904. https://www.mdpi.com/2227-7390/10/16/2904.

Sánchez, J. A., Flores-Rivera, J. A. & Prietch, S. S. (2023, October). Designing a sign language training platform for hearing healthcare personnel. In Proceedings of the

XI Latin American Conference on Human Computer Interaction, 1-9. https://doi.org/10.1145/3630970.3631057.

Sincan, O. M. & Keles, H. Y. (2020). Autsl: A large scale multi-modal turkish sign language dataset and baseline methods. IEEE, 8, 181340-181355.10.1109/ACCESS.2020.3028072.

Sosa-Jiménez, C. O., Ríos-Figueroa, H. V. & Solís-González-Cosío, A. L. (2022). A Prototype for Mexican Sign Language Recognition and Synthesis in Support of a Primary Care Physician. IEEE, 10, 127620-127635. 10.1109/ACCESS.2022.3226696

Trujillo-Romero, F. & García-Bautista, G. (2023). Mexican Sign Language Corpus: Towards an automatic translator. ACM Transactions on Asian and Low-Resource Language Information Processing. https://doi.org/10.1145/3591471.

Varela-Santos, H., Morales-Jiménez, A., Córdova-Esparza, D. M., Terven, J., Mirelez-Delgado, F. D. & Orenday-Delgado, A. (2021). Assistive device for the translation from Mexican sign language to verbal language. Computación y Sistemas, 25(3), 451-464. https://doi.org/10.13053/cys-25-3-3459.

Villanueva, J. G., Islas, L. J. O. & Ramírez, C. I. H. (2023). Experiencias de docentes sobre la enseñanza de la Lengua de Señas Mexicana como lengua natural en personas sordas desde una perspectiva de género. Revista de psicología de la Universidad Autónoma del Estado de México, 12(30), 217-249. https://doi.org/10.36677/rpsicologia.v12i30.20983.

Wen, T. H., Vandyke, D., Mrksic, N., Gasic, M., Rojas-Barahona, L. M., Su, P. H. & Young, S. (2016). A network-based end-to-end trainable task-oriented dialogue system. arXiv preprint arXiv:1604.04562.

Zepeta, H. Z., Rosales, G. A. G., Santiago, H. J. J. & Lee, M. M. (2022). Métricas de rendimiento para evaluar el aprendizaje automático en la clasificación de imágenes petroleras utilizando redes neuronales convolucionales. Ciencia Latina Revista Científica Multidisciplinar, 6(5), 4624-4637. https://doi.org/10.37811/cl_rcm.v6i5.342

CÓMO CITAR ESTE ARTÍCULO:

Caballero Hernández, H., Muñoz Jiménez, V. y Ramos Corchado M, A. (2025). Desarrollo de Software dedicado a la traducción de la Lengua Mexicana de Señas mediate *Deep Learning* y *Machine Learning. Paakat: Revista de Tecnología y Sociedad, 15*(28). http://dx.doi.org/10.32870/Pk.a15n28.897

*Héctor Caballero Hernández es doctor en Ciencias de la Ingeniería por la Universidad del Estado de México en 2020. Sus temas de investigación son esteganografía, sistemas biométricos y computación científica basada en lenguaje natural.

**Vianney Muñoz Jiménez es profesora investigadora en Procesamiento de Imágenes y Visión Computacional en la Universidad Autónoma del Estado de México. En 2009 recibió su doctorado en la Universidad París 13, Francia. Su trabajo de investigación se centra en visión computacional, procesamiento de imágenes, compresión de video, etc.

***Marco Ramos Corchado es profesor investigador en Inteligencia Artificial y Realidad Virtual en la Universidad Autónoma del Estado de México. Obtuvo su doctorado en la Universidad de Toulouse en 2007, Francia. Sus temas de investigación son: Vida Artificial, técnicas de animación, sistemas distribuidos, agentes inteligentes, etc.