Tecnologías de reconocimiento de manos para fomentar la inclusión: interpretación automática del lenguaje de señas chilena

Hand recognition technologies to promote inclusion: automatic interpretation of Chilean sign language

Ismael Rojas Flores[1] https://orcid.org/0009-0005-4029-1361
Mauricio Mamani Chambi[1] https://orcid.org/0009-0000-2670-5322
Diego Aracena Pizarro[1*] https://orcid.org/0009-0002-9182-7654

Recibido 09 de diciembre de 2024, aceptado 15 de mayo de 2025
Received: December 09, 2024 Accepted: May 15, 2025

RESUMEN

Existen personas que sufren, ya sea problema de audición o del habla, ocasionando un problema de comunicación innegable en sus vidas. Estas personas que sufren de esta discapacidad se logran comunicar a través de la utilización del lenguaje de señas, sin embargo, este lenguaje no es muy conocido para una gran parte de la población humana que utilizan el habla o se comunican a través de la escritura. Este Trabajo se centra en el uso de tecnologías de reconocimiento de manos para interpretar el lenguaje de signos adoptando el lenguaje de señas chileno, utilizando técnicas de visión computacional y Machine Learning (ML), con distintas librerías de reconocimiento de objetos, las cuales se utilizaron junto con un conjunto de imágenes 2D de manos, las cuales se prepararon en un archivo CSV. También se utilizó Mediapipe para detectar puntos clave en las manos, logrando una alta precisión en la detección de gestos.

Para mayor efectividad se utilizó OpenCV, Roboflow y Mediapipe Hands para detectar puntos característicos en las manos para detectar la mano (izquierda o derecha) y realizar procedimientos en ella. El módulo especializado Mediapipe Hands del framework Mediapipe que utiliza modelos pre-entrenados de red neuronal convolucional para identificar 21 puntos clave en las manos, permitiendo la detección de gestos y manos en tiempo real, siendo necesarios para detectar una mano y trabajar con ellos en imágenes estáticas o video. OpenCV se empleó para capturar imágenes por segundo a color RGB y renderizar imágenes estáticas. Roboflow se utilizó para entrenar el dataset para el reconocimiento de lengua de señas chilena.

Finalmente, este conjunto de herramientas permitió lograr una aplicación de educación inclusiva en la detección y reconocimiento de señas realizadas con las manos y crear un ambiente de aprendizaje, entre otras posibilidades, de perfección e interpretación de las señas del punto de vista dactilográfico.

Palabras clave: Lenguaje de señas, Mediapipe, Mediapipe Hands, red neuronal convolucional (CNN), OpenCV.

ABSTRACT

There are individuals who endure hearing or speech impairments, resulting in undeniable communication challenges in their lives. These individuals can communicate through sign language; however, this language is not well-known to a large part of the population that uses verbal or written means of

[1] Universidad de Tarapacá. Facultad de Ingeniería. Departamento de Ingeniería en Computación e Informática. Arica, Chile. Email: ismael.rojas.flores@alumnos.uta.cl; mauricio.mamani.chambi@alumnos.uta.cl; daracena@academicos.uta.cl
* Autor de correspondencia: daracena@academicos.uta.cl

CC BY

DOI:10.64966/ingeniare.v33.7

communication. This work focuses on using hand recognition technologies to interpret sign language, specifically adopting Chilean Sign Language, by employing computer vision techniques and Machine Learning (ML) with various object recognition libraries. These were utilized alongside a dataset of 2D hand images, organized into a CSV file. Mediapipe was also used to detect key points on the hands, achieving high accuracy in gesture detection.

OpenCV, Roboflow, and Mediapipe Hands were used to detect characteristic points on the hands, thus identifying the left or right hand and allowing processing to continue. The specialized Mediapipe Hands module within the Mediapipe framework uses pre-trained convolutional neural network models to identify 21 key points on the hands, facilitating real-time gesture and hand detection, which is essential for detecting hands and working with them in static images or video. OpenCV was employed to capture images per second in RGB color and to render static images. Roboflow was used to train the dataset to recognize Chilean Sign Language.

Ultimately, this set of tools enabled the creation of an inclusive educational application for detecting and recognizing signs made with hands, fostering a learning environment and offering possibilities for refining and interpreting signs from a finger-spelling perspective.

Keywords: Sign language, Mediapipe, Mediapipe Hands, Convolutional neural network (CNN), OpenCV.

INTRODUCCIÓN

La investigación realizada se centra en el uso de tecnologías que permitan detectar puntos característicos en la mano, esto con el fin de poder entender cómo es que se logra realizar la captura de cada punto característico mediante el uso de una herramienta diseñada para realizar esta tarea.

Según el último Censo del Instituto Nacional de Estadísticas (INE), en Chile existen alrededor de 188.857 personas con discapacidad auditiva, lo que equivale al 1,1% de la población total. Sin embargo, es probable que esta cifra no refleja adecuadamente la verdadera magnitud del fenómeno, ya que la discapacidad auditiva presenta una amplia gama de severidades y puede no siempre ser declarada en los censos.

Los datos del Servicio Nacional de la Discapacidad (SENADIS) de 2018 revelan que solo un 0,2% de las personas con discapacidad auditiva ha logrado culminar sus estudios universitarios. Estas cifras evidencian las barreras educativas y laborales que enfrentan las personas sordas en nuestro país, subrayando la necesidad de implementar tecnologías que promuevan su plena inclusión social [1].

El uso de tecnologías que permitan mejorar las distintas labores que pueda realizar una persona son necesarias, ya que permiten dar paso a un mejor rendimiento como a su vez incorporar a personas que mantienen algún grado de discapacidad. Es en este sentido que se hace mención a tecnologías como la visión por computación las cuales pueden ayudar a la comunidad a mejorar y ayudar con las distintas labores que se le presenten en el día a día [2].

Siguiendo una metodología del empleo de técnicas de visión se captura los puntos característicos de las manos se hace uso del módulo Mediapipe Hands [3], la cual nos brinda soluciones adecuadas para la investigación, ya que proporciona modelos pre entrenados mediante redes neuronales convolucionales (CNN), diseñadas para procesar imágenes, que tienen una estructura de rejilla. Aplican capas convolucionales para detectar características como bordes y texturas, y son capaces de identificar patrones complejos a partir de elementos simples. Estos modelos permiten enfocarse directamente en puntos característicos de la mano, esto nos desliga del uso de algoritmos para poder detectar los puntos, ya que cuenta con estas funcionalidades internamente. Mediapipe Hands permite identificar 21 puntos característicos necesarios [4] para poder detectar una mano y poder trabajar con ellos ya sea sobre imágenes estáticas o video.

Una vez detectados los puntos característicos de la mano, se procede a normalizar sus coordenadas

para establecer un marco de referencia consistente. Esto implica transformar las coordenadas a un rango específico, eliminando las variaciones causadas por la posición de la mano o la cámara. La información normalizada de los puntos característicos se utiliza como entrada para un algoritmo de reconocimiento de gestos. El resultado del reconocimiento de gestos se traduce al lenguaje natural, generando una salida textual que representa el significado del gesto realizado. Esta traducción se puede presentar visualmente en una interfaz de usuario o mediante síntesis de voz con Pyttsx3 [5].

Para mejorar la accesibilidad y facilidad de uso, se convierte el script de Python en un archivo ejecutable. Esto permite que el sistema pueda ser utilizado por personas sin necesidad de conocimientos técnicos o instalación de software adicional. Con ello, se facilita la comunicación entre personas oyentes y personas con discapacidad auditiva. Teniendo el potencial de transformar la forma en que estas personas interactúan con el mundo que les rodea, promoviendo la inclusión social y mejorando su calidad de vida.

TRABAJOS RELACIONADOS

Hasanov, Alishzade, Nazimzade, Dadashzade y Tahirov [6] llevó a cabo una investigación sobre un desarrollo híbrido de reconocimiento de palabras y un conjunto de datos dataset para la lengua de señas Azerbaiyán (Azerbaijani) [6]. Ellos se plantearon un desafío de crear modelos desde cero sobre un lenguaje complicado y de pocos datos en visión computacional, esto para demostrar la potencialidad de los sistemas y modelos de arquitectura para demostrar la potencialidad y exactitud de los sistemas y modelos de arquitectura.

Primero empezaron en conseguir información con las personas de Azerbaiyán creando un bot de Telegram llamado JestDiliBot donde permitía a los usuarios seleccionar una letra y luego recibir una imagen o video de muestra basado en su selección, luego con la ayuda del usuario, sugería contribuir con las letras que tenían la menor colección de datos.

Abordaron varios desafíos en el reconocimiento de la lengua de señas, como la verificación de la optimización de las variaciones en el deletreo de los dedos, las letras estáticas, dinámicas y la selección eficiente de características para aumentar la eficiencia del modelo propuesto, teniendo arquitecturas tales como las máquinas vectores de soporte (SVM), regresión logística y redes neuronales profundas (DNN); ambos unidos a Mediapipe.

Sus resultados finales incluyeron detalles sobre la selección de características clave para la detección de gestos utilizando el método de selección de características, mostrando la potencia de Mediapipe del modelo Mediapipe + Cluster DNN con una precisión del 94%, resaltando la importancia de una adecuada selección de características y el uso de imágenes de alta resolución.

También se cita el trabajo de El Zaar, Benaya y El allati [7], donde presentan una arquitectura deep learning de alto rendimiento basada en la red CNN [7] para el reconocimiento de lenguaje de señas. El enfoque propuesto logró tasas de reconocimiento del 99% para la lengua de signos americana (ASL) y los alfabetos de signos irlandeses (ISL), y del 98% para el alfabeto de la lengua de signos árabe (ArASL), superando a los métodos más modernos, destacando la potencia del modelo CNN y deep learning. La arquitectura, basada en el modelo VGG, incluye capas convolucionales, capas de agrupación y capas completamente conectadas, entrenadas y probadas en múltiples conjuntos de datos de lenguaje de señas, lo que muestra resultados prometedores para mejorar las vidas de las personas sordas.

Nuestro estudio propone un nuevo enfoque para el reconocimiento de la Lengua de Señas Chilena (LSC). A diferencia de los trabajos previos, esta investigación se centra específicamente en optimizar la detección y reconocimiento de gestos de la LSC en tiempo real, con el objetivo de mejorar la inclusión educativa para la comunidad sorda chilena. Siguiendo las recomendaciones de los trabajos relacionados, utilizaremos Mediapipe por su eficacia en la detección de gestos y su potencia en las redes neuronales convolucionales.

DEFINICIÓN DEL PROBLEMA A SOLUCIONAR

Muchas personas en el mundo enfrentan dificultades auditivas o del habla, lo que genera una barrera comunicacional innegable en sus vidas. Para superar esta barrera, han desarrollado la lengua de señas,

un lenguaje visual que les permite expresarse y comunicarse de manera efectiva. Sin embargo, este lenguaje no es ampliamente conocido por la población en general, lo que limita las oportunidades de interacción e inclusión social de las personas sordas.

Conscientes de esta problemática, se propone desarrollar una aplicación de enseñanza de la lengua de señas chilena. Este proyecto busca ser accesible y fácil de usar, eliminando barreras tecnológicas y educativas que impiden el aprendizaje efectivo de esta lengua.

El proyecto se encuentra en una etapa avanzada de desarrollo. Se realizan labores de depuración, documentación y mejora para garantizar una experiencia de aprendizaje óptima. Un aspecto fundamental es la creación de un ejecutable que permita a las personas utilizar el proyecto sin necesidad de conocimientos técnicos o de instalar programas complejos.

El Departamento de Informática está firmemente comprometido con la inclusión, entregando el desarrollo de soluciones tecnológicas que beneficien a las personas con discapacidad. Este proyecto se enmarca en este compromiso y busca brindar una herramienta útil y accesible para la comunidad sorda.

Se espera que este proyecto tenga un impacto significativo en la comunidad sorda, facilitando el acceso al aprendizaje de la lengua de señas chilena y promoviendo su inclusión social. A través de este proyecto, se busca derribar barreras comunicacionales y fomentar una sociedad más justa e inclusiva para todas las personas, Figura 1.

RECONOCIMIENTO DE MANOS Y LENGUAJE DE SEÑAS

El reconocimiento de gestos por cámara es una capacidad de las computadoras y dispositivos para encontrar e interpretar diferentes tipos de gestos humanos capturados en la cámara. Estos gestos pueden identificar símbolos escritos con los dedos y movimientos dinámicos de las manos donde capturan los gestos y hacen el uso de algoritmos de aprendizaje automático para analizar e interpretar los datos capturados. El reconocimiento de gestos ofrece formas innovadoras de interactuar con interfaces, como controlar un dispositivo o reproducir música mediante movimientos específicos.

ENTRENAMIENTO

Para el modelo de entrenamiento de la lengua de señas chilena, se implementó un proceso de entrenamiento que busca identificar la palabra correspondiente a la señal capturada en el video de la cámara, como se ve en la Figura 2. Todo este proceso tiene como objetivo concientizar y facilitar el aprendizaje de la lengua de señas chilena mediante el uso de visión computacional.

Como uso de herramienta práctica se recomienda el uso de la plataforma de “Roboflow” [8]. A grandes rasgos, la plataforma de Roboflow sirve para crear dataset enfocados en el proyecto. En cierto modo es lo más acorde para poder continuar con el trabajo de detección de las manos y darle un enfoque a la lengua de señas, ya que esta plataforma permite asignar las clases a cada una de las imágenes, siendo las clases cada una de las letras de los abecedarios en la lengua de señas chilena. luego permite exportar dicho dataset en un formato adecuado para su trabajo posterior de entrenamiento, Figura 3, Figura 4 y Figura 5.

Para el uso de la cámara, se optó por emplear OpenCV [9], la cual es una librería de computación visual de procesamiento de imágenes de Python. Esto nos ayudará a capturar frames por segundo a color RGB, donde con ayuda de Mediapipe se obtiene la detección de gestos y manos.

Fuente: Elaboración propia.



Figura 1. Flujo del proyecto.

Fuente: [7]

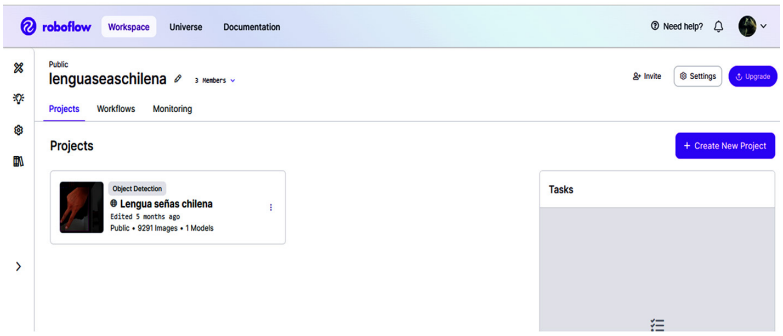


Figura 2. Proyecto creado en Roboflow.

Fuente: [7]

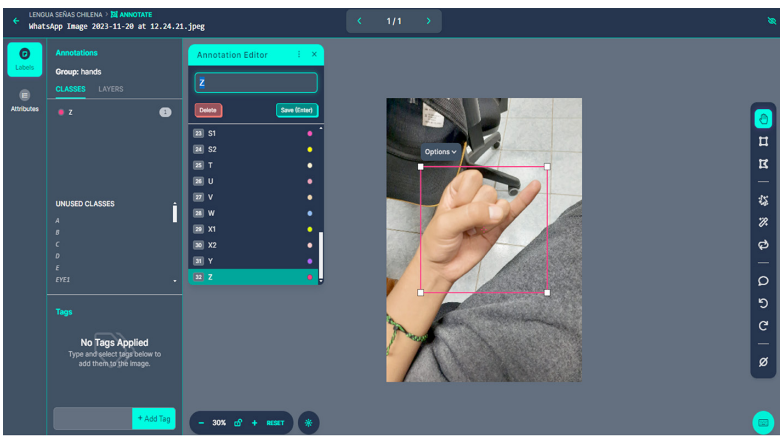


Figura 3. Asignación de clase a la imagen subida.

Fuente: [7]

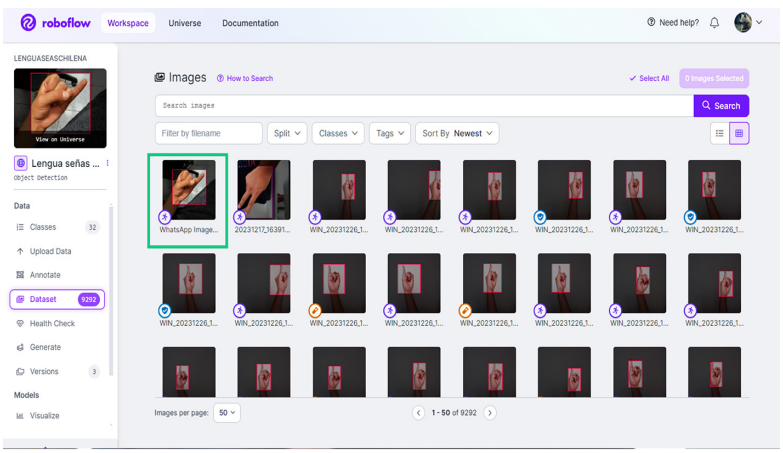


Figura 4. Nueva imagen en el dataset.

Fuente: [7]

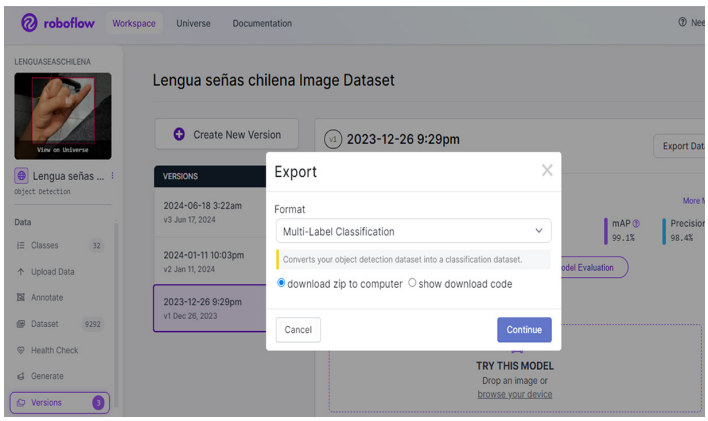


Figura 5. Configuración de exportación del DataSet.

OpenCV también ayudó en la renderización de las imágenes estáticas que están extraídas en las imágenes del dataset de lengua de señas chilenas.

PROCESAMIENTO Y SALIDA

Para detectar las manos, se utiliza el framework "Mediapipe" debido a que el trabajo relacionado recomienda utilizarlo, además de que su resultado es rápido y proporciona una representación precisa en el campo de detección de cámaras. Mediapipe no sólo calcula la posición de las manos, sino que también identifica los dedos y las yemas de los dedos y las palmas de las manos, estimando los 21 puntos clave de la mano que el framework realiza [10].

Mediapipe entrega soluciones de aprendizaje automático en el sistema, fáciles y accesibles. Con éste se extraen todos los puntos de referencia clave de las manos. También proporciona un conjunto de bibliotecas y herramientas que facilitan la aplicación de técnicas de inteligencia artificial y aprendizaje automático en diversas plataformas de desarrollo. Como se muestra en la Figura 6, donde el framework ofrece varias soluciones que involucran varios modelos.

Fuente: [2]

Solución	Android	Web	Python	iOS	Personalizar modelo
API de inferencia de LLM	●	●		●	●
Detección de objetos	●	●	●	●	●
Clasificación de imágenes	●	●	●	●	●
Segmentación de imágenes	●	●	●		
Segmentación interactiva	●	●	●		
Detección de puntos de referencia manuales	●	●	●	●	
Reconocimiento de gestos	●	●	●	●	●
Incorporación de imágenes	●	●	●		
Detección de rostro	●	●	●	●	
Detección de puntos de referencia facial	●	●	●		
Estilización de rostros	●	●	●		●
Detección de puntos de referencia de posiciones	●	●	●		
Generación de imágenes	●				●
Clasificación de texto	●	●	●	●	●
Incorporación de texto	●	●	●		
Detector de lenguaje	●	●	●		
Clasificación de audio	●	●	●		

Figura 6. Soluciones de Mediapipe.

En este caso, se utilizó "Reconocimiento de gestos" la cual se conoce como "Mediapipe Hands", solución diseñada para la detección en tiempo real de manos y sus puntos clave por cámara. Lo primero que realiza es la detección de palma, mediante la detección integrada de una región de interés (ROI) identifica la palma de la mano. Luego se aplica un modelo de red neuronal convolucional (CNN) donde recibe la ROI y estima los 21 puntos clave de la mano, como se muestra en la Figura 7, estos puntos incluyen las posiciones de las articulaciones de los dedos y otras características anatómicas clave. Estos puntos se denominan "landmarks".

Todas estas características se logran con la librería "landmarkHands", gracias a esto, permite detectar apropiadamente las palmas de la mano en diferentes iluminaciones y poses, siendo capaz de manejar múltiples manos en una sola imagen. En el proceso, la integración de estos sistemas en el reconocimiento de lenguas de señas no solo mejora la precisión y la robustez de los sistemas de traducción, sino que también abre nuevas posibilidades para la educación inclusiva y la interacción natural con la tecnología.

ARQUITECTURA DEL SISTEMA

La arquitectura del sistema presentada en la Figura 8, se centra en el uso de procesos teniendo como núcleo la detección de las manos sea para encontrar la seña o la mano. Este proceso requiere el uso de los datos de la cámara y la lengua de señas almacenadas en una base de datos, donde el sistema interactúa con la cámara para obtener los datos necesarios para el reconocimiento de la mano.

1. Entrada
   a. Señales de mano: Señal de la mano en la cámara.
   b. Cámara web: captura de imágenes de las manos del usuario en tiempo real.
2. Iniciación
   a. Imágenes de lengua de señas chilena: Imágenes que buscan entrenar el modelo.
   b. Base de datos: Entrada de datos es la base de datos de los lenguajes de señas chilenos, utilizada para el entrenamiento y creación del modelo correspondiente
3. Detección
   a. Extracción de características: OpenCV se utiliza para extraer características de las imágenes preprocesadas, como la forma, la orientación y la textura de las manos, luego con Mediapipe Hands se utiliza para detectar los puntos clave de las manos del usuario en las imágenes capturadas.
   b. Carga de datos de aprendizaje: Carga el modelo guardado en el proceso de reconocimiento de manos.
   c. Normalización de las características: Se normalizan valores para las coordenadas (x, y).
   d. Identificación de la señal emitida: Una vez hecho el cálculo de normalización, se detecta la señal emitida por la cámara web.
4. Salida
   a. Salida de puntos característicos: Se extrae el punto característico final y se clasifica la mano.
   b. Entrega de la palabra de la señal: El sistema genera una salida que indica la palabra o frase de la lengua de señas reconocida.

Fuente: [3]

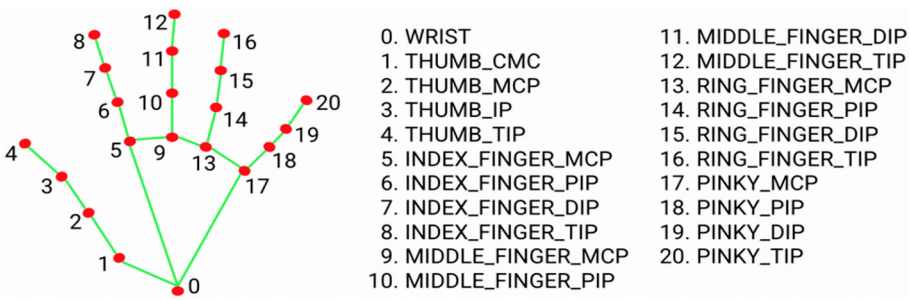


Figura 7. 21 puntos clave CNN.

Fuente: [Propia]

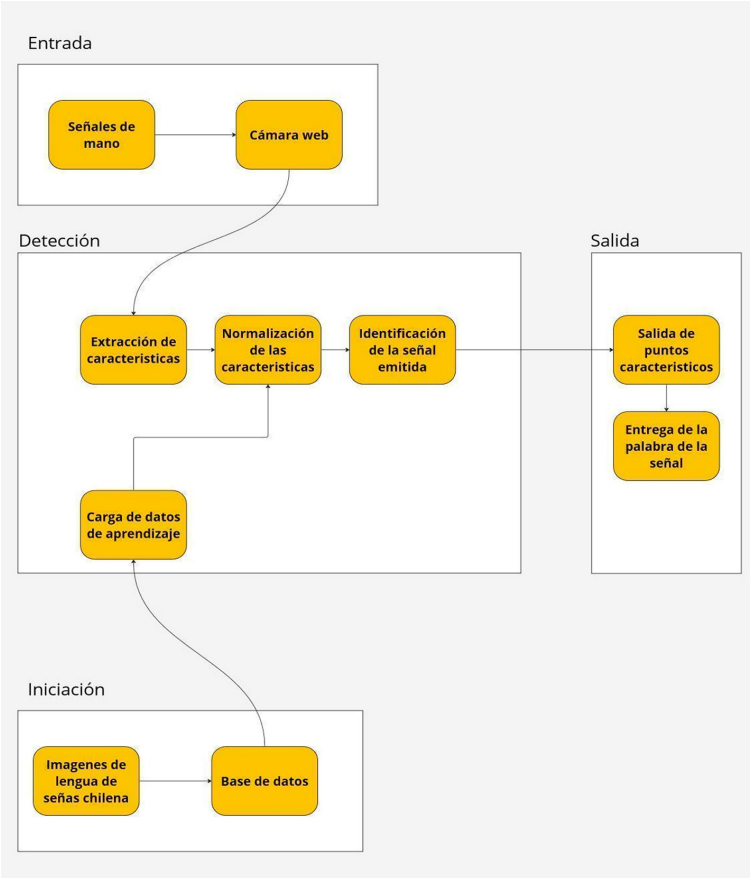


Figura 8. Arquitectura del sistema.

A continuación, en la Figura 9 se presenta un diagrama BPM, que detalla los procedimientos y las secuencias de operaciones dentro del sistema.

Por último, se incluye un diagrama de bloques que resume los elementos clave y sus conexiones esenciales, Figura 10.

PROCESO DE IMPLEMENTACIÓN

Se han realizado uso de las siguientes herramientas y librerías de python:

1. *Numpy*: Para operaciones que involucran la manipulación de arreglos [11].
2. *OpenCV*: Captura de imágenes de la cámara web y procesamiento de imágenes python;
3. *Keras*: Para el procesamiento y clasificación de los datos con red neuronal [12].
4. *Mediapipe Hands*: Identificación de las articulaciones y puntas de los dedos de las manos;
5. *PyQt5*: Para la creación de la interfaz gráfica del modelo de detección;
6. *Argparse*: Procesamiento de argumentos de línea de comandos [13].
7. *Deque*: Almacena temporalmente datos o procesarlos de manera eficiente [14].
8. *Itertools*: Creación de iteradores, filtrado de datos, agrupamiento y otras operaciones [15].
9. *Pyttsx3*: Permite la conversión de texto a voz;
10. *TensorFlow*: Biblioteca para la creación de modelos de aprendizaje automático [16].
11. *Pyinstaller*: Librería de python la cual nos permite agrupar aplicación junto con las

Fuente: Elaboración propia

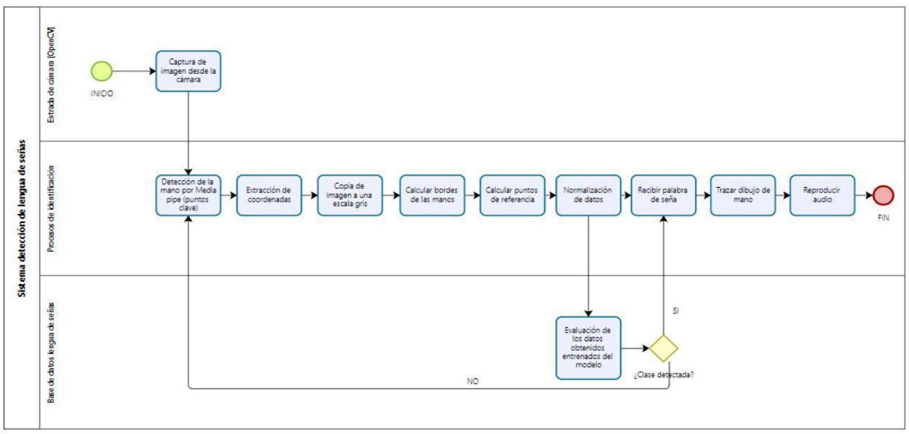


Figura 9. Diagrama BPM.

Fuente: [Propia]

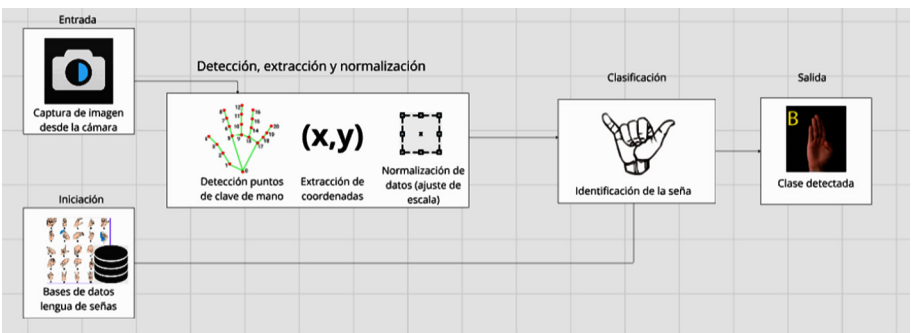


Figura 10. Diagrama simplificado de bloques.

dependencias que esta requiera en un solo paquete, de esta forma los usuarios pueden ejecutar la aplicación sin necesidad de un intérprete del lenguaje Python ni ningún módulo extra [17].

En la Figura 11 se muestra el funcionamiento del sistema.

INTERFAZ GRÁFICA PARA LENGUAJE DE SEÑAS

Antes de todo, se crea la interfaz gráfica del usuario utilizando PyQt5, aquí se obtiene la resolución de la pantalla del usuario y se configuran varios elementos de la interfaz, como el título de la ventana, el color de fondo, la ubicación de una imagen, y la creación de botones para diferentes acciones: "Clases" que llevan a las clases de lengua de señas, "Prueba" que activa la cámara y detecta las señas de las manos, y finalmente "Salir" que cierra la interfaz. La ventana principal presenta una imagen centrada acompañada de botones que ostentan efectos de sombreado, estilos CSS puros y bordes personalizados. Estos botones, al ser presionados, permiten la interacción del usuario para navegar por la aplicación, la cual se muestra en pantalla completa, Figura 12.

Debido al enfoque de la investigación en la detección de manos y lengua de señas, no se detallarán las funcionalidades de la aplicación más allá de este punto. Sin embargo, es importante destacar que aquí es donde comienza el sistema.

Fuente: Elaboración propia.

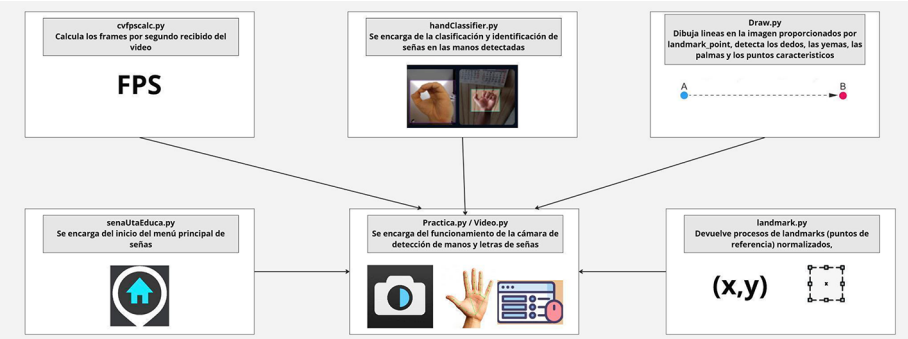


Figura 11. Implementación del sistema.

Fuente: Elaboración propia.

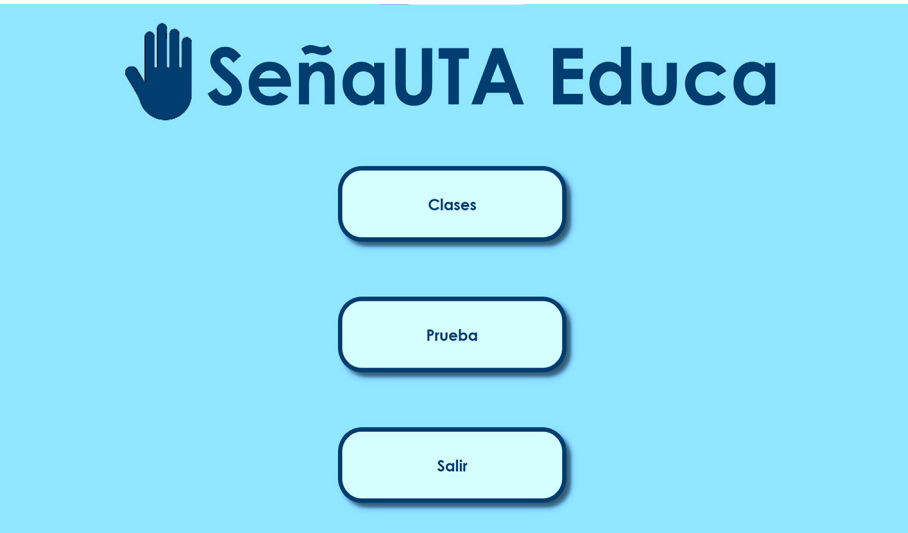


Figura 12. Interfaz inicial.

DESARROLLO DEL SISTEMA DE RECONOCIMIENTO DE SEÑAS

En el código de práctica, ya sea “practica.py” o “video.py”, se inicializa la interfaz del modelo utilizando “argparse” para definir y parsear argumentos de línea de comandos. Estos argumentos incluyen el ancho y alto de la captura, un modo de imagen estática y niveles de confianza mínimos para la detección y el seguimiento. Con “OpenCV”, se captura la imagen de la cámara y se define la resolución del video. Luego, se configura Mediapipe Hands para detectar y seguir dos manos en el video con parámetros específicos.

Se carga un modelo de clasificación de lengua de señas chilena desde “handClassifier.py”, utilizando TensorFlow Lite para realizar inferencias. Las etiquetas de las clases incluyen las letras de la A a la Z y sus coordenadas respectivas, considerando también las letras dinámicas. Se encuentra la clase con mayor probabilidad en el tensor de salida, basándose en los datos de entrada proporcionados.

Luego, volviendo al archivo practica.py/video.py, se procede a realizar un bucle para el funcionamiento del video. Se obtiene la medición de FPS mediante un archivo llamado “cvpscalc.py”. Recibiendo estos fotogramas por segundo, se utiliza Mediapipe Hands

y OpenCV para procesar la imagen. Esto incluye la conversión de BGR a RGB y la creación de una copia de imagen en blanco y negro para mayor precisión.

La función "landmark" por el archivo "landmark.py" normaliza los puntos de referencia de las manos, calculando el rectángulo delimitador y creando una lista de coordenadas normalizadas. La razón por la cual se realiza este proceso es porque las coordenadas de dos imágenes del mismo gesto pueden ser diferentes. Por esta razón, se normalizan las coordenadas estableciendo las coordenadas a (0,0). Lo primero que se hace es calcular los bordes de las manos utilizando las landmarks de Mediapipe.

Una vez normalizadas, se determina la seña utilizando el archivo "hand_classifier", donde se guardó el modelo de clasificación por handClassifier.py. Los puntos característicos y los bordes se trazan con "draw.py", representando las palmas y los dedos en la imagen.

Obtenidos estos datos en practica.py, junto con la etiqueta de la clase de la seña detectada, se guarda la letra para su posterior visualización. En caso de variaciones de letra estática o dinámica, es necesario esperar algunos segundos adicionales. Finalmente, se muestra la letra detectada y se ejecutan acciones en base a ciertas condiciones, como mostrar "CORRECTO" y utilizar la función "speak(text)" para vocalizar el carácter detectado. Todo el procedimiento, desde la recepción de FPS hasta la recepción de la seña y su vocalización, se lleva a cabo en un único fotograma. Para finalizar, se colorea la imagen en blanco y negro, se transforma el fotograma a formato QImage (PyQt5) y, por último, se emite la señal para redimensionar la cámara y renderizar el texto.

Después de aplicar los archivos del sistema de reconocimiento de señas, se ha demostrado la capacidad del sistema para capturar correctamente la señal y reconocer la letra correspondiente. La Figura 13 y Figura 14 muestra la cámara en acción, mostrando cómo el sistema detecta la señal y muestra la letra correspondiente en tiempo real.

ESCENARIO

Como escenario se optó por realizar una clase con alumnos que cursan el curso de E.F.G lengua de señas, esto con el fin de darle contexto al desarrollo de la aplicación educativa de lengua de señas. Esta clase trataba de introducir a los estudiantes de manera simple sobre el funcionamiento de la aplicación mediante una presentación, estableciendo la problemática y solución. Luego, se mostraba la funcionalidad de la aplicación, posterior a esta explicación los estudiantes tenían a disposición cinco computadoras portátiles con la aplicación cargada y lista para poder hacer uso de ella. Cada estudiante podía utilizar la aplicación a libre uso,

Fuente: Elaboración propia.



Figura 13. Evidencia de la implementación.

Fuente: Elaboración propia.

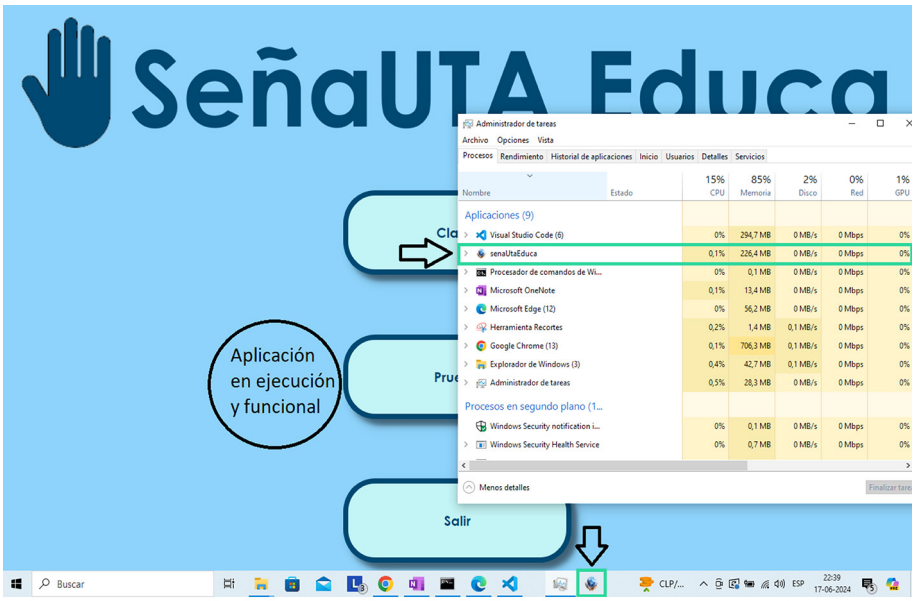


Figura 14. Aplicación SeñaUtaEduca en ejecución.

en tal caso de haber algún tipo de complicación se acudía a dar ayuda en el uso de la aplicación. Al finalizar la clase se realizaban consultas sobre el funcionamiento de esta y si es que hubo complicación en su uso.

El objetivo de este escenario es poder evidenciar a que nivel de conformidad se puede estar con el uso de esta aplicación, y cómo es que logra ayudar al aprendizaje de los estudiantes sin mucha complejidad, Figura 15.

Fuente: Elaboración propia.



Figura 15. Evidencia escenario 1.

ANÁLISIS DE LOS RESULTADOS

Esta clase contó con la presencia de una docente en Educación Parvularia especializada en la lengua de señas, dos personas sordas que se comunicaban en lengua de señas y el equipo de trabajo, junto con alumnas del curso E.F.G lengua de señas. Finalizada la clase y en el transcurso de esta se pudo evidenciar lo fácil que era utilizar la interfaz de la aplicación. Surgieron algunos cambios que se debían de aplicar al módulo de clases (Lenguaje Dactilográfico), estas modificaciones son más que nada en el apartado visual, ya que había letras que mostraban las señas con ligero grado de inclinación que no era adecuado en la seña que corresponde, algunos problemas en el reconocimiento de letras dinámicas, ya sea por limitaciones de hardware en las cámaras o poca luz en las zonas donde se detectaban las manos.

Finalmente se logra establecer una conversación con el curso sobre el uso de la aplicación, transmitiendo su conformidad con lo que se realizó, fue accesible y fácil de usar, por otra parte, recomendaron seguir mejorando en la aplicación y poder continuar con el módulo de palabras y frases cortas, el cual se considera como trabajo futuro (Tabla 1). En la Figura 17 se observa a las estudiantes del curso de E.F.G de lengua de señas utilizando la aplicación.

Para poder explicar cómo es que cada una de las señas que hacen referencia a una letra del abecedario tiene un gran porcentaje de precisión a la hora de reconocer los gestos de las manos, se debe de explicar que se entrena un modelo con técnicas de inteligencia artificial el cual servirá para poder clasificar cada uno de los gestos del usuario. Los porcentajes de exactitud a la hora de predecir la seña en base a los gestos de la mano son del 98,51%, esto quiere decir que el modelo tiene un buen desempeño a la hora de predecir las señas de las letras del abecedario. En la Figura 16 se puede observar la evaluación del modelo entrenado con anterioridad, en esta evaluación se observa la precisión categórica del modelo a la hora de reconocer los gestos de las manos.

LIMITACIONES

Anteriormente se mencionó una limitación que puede provocar algún tipo de problema con el funcionamiento de la misma, si bien es a nivel de hardware, no deja de ser un punto clave para que el rendimiento mejore a la hora de predecir las señas que proporciona el usuario.

Mediante las pruebas realizadas en el curso E.F.G de lengua de señas se contaba con dos laptops, siendo una de estas la que tenía un mejor desempeño

Tabla 1. Análisis de los resultados.

Participantes	Observaciones	Conclusiones
Profesoras de educación parvularia	Usaban la aplicación junto con una alumna para explicar.	La aplicación les gustó, y esperan que en el futuro se agreguen más módulos.
Personas sordas	Utilizaron la aplicación guiados por los participantes del proyecto.	La aplicación les encantó y la aprobaron, aunque detectaron problemas con las señas “P”, “T” y “X”, y con las señas dinámicas.
Alumnos del curso E.F.G.	Probaron la aplicación y observaron su funcionamiento en diferentes condiciones.	Aprobaron con éxito la aplicación, pero señalaron que en zonas con poca luz no detectaba correctamente y había problemas con las señas dinámicas.

Fuente: Elaboración propia.

Fuente: Elaboración propia.

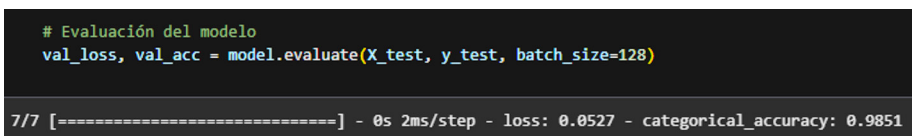


Figura 16. Precisión del modelo entrenado.

Fuente: Elaboración propia.



Figura 17. Evidencia escenario 2.

a la hora de reconocer las señas, no era mucha la diferencia ya que ambas laptops lograban reconocer las señas pero en temas de poca luminosidad la laptop del modelo "idepad s540-13IML", marca "Lenovo" podía reconocer señas en zonas con poca luminosidad mejorado la experiencia de los usuarios, ya que no presentaron mayores complicaciones en la laptop "Lenovo" a la hora de probar los gestos de las manos con la aplicación.

Otra de las limitaciones que se debe de discutir es acerca de las señas dinámicas, si bien se da una solución simple para poder trabajar con señas dinámicas, esta solución logra su objetivo, pero presenta un desafío para los usuarios el poder realizarlas. Las señas dinámicas involucran no solo las manos, sino también las expresiones faciales y el cuerpo. Capturar y comprender la sincronización precisa de estos elementos puede ser complicado para un modelo de inteligencia artificial. La solución que se implementó es lo más cercano a una solución futura, ya que, si bien es simple, trata de ejemplificar cómo es que trabajando en un modelo multidimensional se podría lograr entrenar a mayor precisión los gestos dinámicos, esto subdividiendo las clases de letras dinámicas en subclases las cuales mantienen su propio conjunto de imágenes y por lo cual cada una de estas subclases que componen a una letra dinámica pueden ser llamadas una a una por el clasificador en un orden acorde, y con ello reconocer punto a punto dicha seña, de esta forma el modelo podría brindar soporte a la señas dinámicas y con ello mejorar la experiencia de los usuarios. Cabe mencionar que esta solución implica aumentar el tamaño del conjunto de datos, esto incrementa la complejidad del sistema, tanto en entrenamiento como en inferencia, debido al crecimiento del conjunto de datos y la necesidad de procesar secuencialmente las subclases dinámicas. Dependiendo de K (el número de subclases), la complejidad podría aumentar a $0\,(n \cdot k)$, (n es el número de clases dinámicas y k el número de subclases), tanto en términos de datos como de inferencia. Esto implica mayor tamaño del dataset y por lo cual mayor procesamiento para poder entrenar el modelo.

PERCEPCIONES DEL ESCENARIO

El equipo de trabajo observó un entusiasmo considerable por parte de los estudiantes al utilizar la aplicación, lo cual indicaba una novedad en su experiencia de aprendizaje. Como se sabe, los estudiantes recurrían a libros, videos o artículos en línea para obtener información sobre la lengua

de señas. La aplicación, al ser una herramienta interactiva, facilitó el aprendizaje de una manera más lúdica, ya que los resultados sugieren que la aplicación ofrece un medio de aprendizaje inclusivo y de fácil uso. Además, se recopilaron diversas sugerencias para mejorar la aplicación, las cuales serán fundamentales para garantizar su aceptación a largo plazo. La Figura 18 muestra la pantalla de introducción de la aplicación, diseñada para facilitar su uso por parte de los estudiantes.

CONCLUSIÓN

Se convirtió el script de Python en un archivo ejecutable, mejorando la accesibilidad y facilidad de uso para usuarios sin conocimientos técnicos y se normalizaron las coordenadas de los puntos característicos de la mano para establecer un marco de referencia consistente. Esto elimina las variaciones causadas por la posición de la mano o la cámara, asegurando un reconocimiento de gestos preciso. El desarrollo de este sistema tiene un gran potencial para impactar positivamente en la vida de las personas con discapacidad auditiva. Facilita la comunicación entre personas oyentes y personas con discapacidad auditiva, promoviendo la inclusión social y mejorando la calidad de vida.

Se logra hacer la unión de la plataforma Roboflow como también de las librerías necesarias para esta aplicación y de esta forma tenerla lo más completa posible. Se da una breve introducción a la plataforma de Roboflow, si bien es simple la explicación, se puede escalar a más y poder dar a entender cómo es que se crea el dataset del proyecto, este proceso si bien es bastante rutinario a la hora de agregar nuevas imágenes al dataset fue un desafío en un inicio, ya que se contaba con un dataset externo, el cual no tenía las señas correctas que uno esperaba sobre la lengua de señas chilena, es por ello que se decidió crear el dataset desde cero con el fin de poder tener el dataset lo más correcto posible, para que no hubieran fallo a la hora de reconocer las señas. Luego se da una explicación bastante detallada de los módulos que componen la aplicación, los cuales son necesarias para tener una correcta funcionalidad de la aplicación. Y por último se explica cómo fue el proceso para poder empaquetar la aplicación y poder tenerla en un formato más amigable para el público en general.

La aplicación se presentó a un curso E.F.G. de lengua de señas, donde se recibió una respuesta positiva y se generó un gran interés entre los asistentes. Los usuarios apreciaron la propuesta y destacaron la utilidad potencial de la herramienta. Sin embargo, algunos usuarios también mencionaron algunas áreas de mejora: las letras dinámicas eran difíciles de realizar y que a veces la detección fallaba en condiciones de poca luz. A pesar de

Fuente: Elaboración propia.



Figura 18. Evidencia escenario 3.

estas observaciones, la mayoría de los participantes valoraron el proyecto de manera positiva y expresaron su deseo de ver su desarrollo continuo. También, una persona con discapacidad que se comunicó con lengua de señas, probó la aplicación y les gustó.

La presentación de la aplicación a un curso de lengua de señas ha sido una experiencia exitosa, con resultados positivos y un gran interés por parte de los asistentes. Los comentarios recibidos han sido valiosos para identificar áreas de mejora y establecer prioridades para el desarrollo futuro de la aplicación.

Como trabajo futuro, se busca implementar la capacidad de interpretar frases en la aplicación de lengua de señas. Esta nueva funcionalidad la llevaría a un nivel más avanzado y la convertiría en una herramienta aún más útil para la comunidad sorda.

AGRADECIMIENTOS

Gracias a la colaboración de Cristian Fritis Palacios, José Díaz Paz y Katia Layi Fernandez, en el curso de Visión Computacional, donde se sentaron las bases iniciales de este proyecto. Posteriormente, en una segunda etapa los integrantes, Patricio Gutiérrez Zuñiga y Katia Layi Fernandez, en un curso de proyecto se logró un avance significativo el proyecto incorporando nuevas funcionalidades relacionadas con el mecanismo de letras dinámicas, un nuevo conjunto de datos y mejorando su usabilidad.

Además, es importante destacar la ayuda proporcionada por la profesora de educación parvularia Vilma Arévalos Lancellotti y su ayudante, del Departamento de educación de la Universidad de Tarapacá. Gracias a su conocimiento se pudo comprender más en profundidad sobre la lengua de señas, un campo ajeno a la formación en Ingeniería Civil en Computación e Informática. Con su apoyo, se pudo corregir el dataset para tener las imágenes lo más correcto posible, y se desarrolló la interfaz con recomendación de las especialistas, lo que permitió desarrollar una interfaz que permitiera un acceso fácil a la aplicación, además se definen colores que no afecten la visión de los usuarios, con títulos y botones llamativos.

REFERENCIAS

[1] M.T. Hidalgo, “La discapacidad auditiva/sordera y ajustes razonables en educación superior”. upla.cl. Accedido: 20 de marzo de 2024. [En línea]. Disponible en: https://www.upla.cl/noticias/PERSONAS-CON-DISCAPACIDAD-AUDITIVA-2.pdf

[2] Enseñas, “¿Qué es la lengua de señas chilena?”, enseña.jgm.uchile.cl. Accedido: 10 de marzo de 2023. [En línea]. Disponible en: https://xn--ensea-rta.jgm.uchile.cl/

[3] Google AI for developers, “MediaPipe Solutions guide,” ai.google.dev. Accessed: Sep. 16,2023. [Online]. Available: https://ai.google.dev/edge/mediapipe/solutions/guide

[4] Google-AI-Edge, “MediaPipe Hands,” mediapipe.readthedocs.io. Accessed: Jul. 13,2024. [Online]. Available: https://mediapipe.readthedocs.io/en/latest/solutions/hands.html

[5] M.S. Sidiq, “Un tutorial esencial de texto a voz de Python utilizando la biblioteca pyttsx3”, hackernoon.com. [En línea]. Disponible en: https://hackernoon.com/lang/es/un-tutorial-esencial-de-texto-a-voz-de-python-usando-la-biblioteca-pyttsx3

[6] J. Hasanov, N. Alishzade, A. Nazimzade, S. Dadashzade, and T. Tahirov, “Development of a hybrid word recognition system and dataset for the Azerbaijani sign language dactyl alphabet,” *Speech Communication*, vol. 53, Art. no. 102960, doi: 10.1016/j.specom.2023.102960.

[7] A. El Zaar, N. Benaya, and A. El Allati, “Sign language recognition: High performance deep learning approach applyied to multiple sign languages,” in *10th International Conference on Innovation, Modern Applied Science & Environmental Studies (ICIES’2022)*, vol. 351, Art. no. 01065, 2022, doi: 10.1051/e3sconf/202235101065.

[8] Roboflow, “Everything you need to build and deploy computer vision applications,” roboflow.com. Accessed: May, 26, 2024. [Online]. Available: https://roboflow.com/

[9] OpenCV team, “OpenCV is the world’s biggest computer vision library,” opencv.org. Accessed: Jun. 01, 2024. [Online]. Available: Disponible en: https://opencv.org/

[10] G. Solano, “Como usar MEDIAPIPE HANDS ? | Python – MediaPipe – OpenCV,” omes-va.com. 2021. [Online]. Available en: https://omes-va.com/mediapipe-hands-python/

[11] T.E. Oliphant, "Guide to NumPy," numpy.org. Accessed: Aug. 25, 2023. [Online]. Available: https://numpy.org/doc/stable/reference/

[12] Keras, "A superpower for ML developers," keras.io. Accessed: Aug. 25, 2023. [Online]. Available: https://keras.io/

[13] Python Software Foundation, "Argparse – Parser for command-line options, arguments and subcommands," docs.python.org. Accessed: Jun. 14, 2024. [Online]. Available: https://docs.python.org/3/library/argparse.html

[14] L. Pozo Ramos, "Python's deque: Implement Efficient Queues and Stacks," realpython.com. Accessed: Jun. 5, 2024. [Online]. Available: https://realpython.com/python-deque/

[15] Python Software Foundation, "Itertools – Functions creating iterators for efficient looping," docs.python.org. Accessed: Aug. 20, 2024. [Online]. Available: https://docs.python.org/3/library/itertools.html

[16] TensorFlow, "Plataforma de extremo a extremo enfocada en el aprendizaje automático", tensorflow.org. Accedido: 20 de agosto de 2024. [En línea]. Disponible en: https://www.tensorflow.org/?hl=es-419

[17] G. Solano, "Como crear ejecutables en Python | PyInstaller (Parte 1)", omes-va.com. 2023. Accedido: 20 de marzo de 2024. [En línea]. Disponible en: https://omes-va.com/como-crear-ejecutables-en-python-pyinstaller-parte-1/