DESARROLLO DE UN MOTOR DE RECONOCIMIENTO DE LA LENGUA DE SEÑAS VENEZOLANA (CASO: INGENIERÍA DE SISTEMAS)

Por

Br. Carlos David Bone Hage

Tutor: Prof. Alejandro Mujica

Desarrollo de un motor de reconocimiento de la lengua de señas venezolana (Caso: Ingeniería de Sistemas)

Br. Carlos David Bone Hage

Proyecto de Grado - Sistemas Computacionales, 47 páginas.

Resumen: El presente proyecto de tesis se enfoca en el desarrollo de un motor de reconocimiento de la Lengua de Señas Venezolana (LSV) con una aplicación específica en el campo de la Ingeniería de Sistemas. Este estudio surge de la necesidad de proporcionar herramientas tecnológicas que mejoren la comunicación y la accesibilidad para las personas sordas, facilitando su inclusión en diferentes ámbitos educativos y profesionales.

La investigación se desarrolla en el contexto de una notable escasez de herramientas adaptadas a la LSV, particularmente en Venezuela, donde la diversidad lingüística y cultural presenta desafíos únicos. Este proyecto no solo busca llenar un vacío importante en el reconocimiento de la LSV, sino que también apunta a la primera aproximación de la creación de una solución adaptada que mejore la comunicación efectiva y sin barreras para los usuarios de la LSV en Venezuela.

La metodología empleada es un enfoque cuantitativo y experimental que utiliza técnicas de Machine Learning y visión por computadora para el desarrollo del motor de reconocimiento. Se optó por el uso de MediaPipe Holistic y redes neuronales profundas para la detección y clasificación de gestos, proporcionando así una base técnica prometedora para el reconocimiento de la lengua de señas. Este enfoque se complementa con un proceso de recopilación y análisis de datos, utilizando videos representativos de la LSV asociados al campo de la ingeniería de sistemas.

Uno de los principales desafíos del proyecto fue la adaptación de tecnologías existentes para el reconocimiento de gestos a las particularidades de la LSV, lo que implicó un significativo esfuerzo en la integración de dichas herramientas. La evaluación del motor desarrollado demostró una alta precisión en el conjunto de gestos abarcado.

El estudio concluye destacando la importancia de continuar con el desarrollo y mejora de tecnologías inclusivas que puedan adaptarse a las variadas y específicas necesidades de las comunidades con discapacidades auditivas. Además, se recomienda la extensión de esta

investigación para abarcar otras variantes de lenguas de señas y otros contextos, lo que podría resultar en una herramienta aún más versátil y accesible.

Palabras clave: Lengua de señas, Inteligencia Artificial, Redes Neuronales, MediaPipe Holistic, Computer Vision.

Índice

Capítulo 1

Introducción

Las actividades más comunes en nuestras vidas, aquellas que damos por sentado, pueden resultar enigmáticas cuando tratamos de entenderlas sistemáticamente [1]. La comunicación es la columna vertebral de la experiencia humana, un proceso fundamental que nos conecta, nos informa y nos une como sociedad. Sin embargo, para millones de personas con discapacidad auditiva, la comunicación efectiva se ve obstaculizada por la barrera del lenguaje. La incapacidad para entender y ser entendido puede crear una desconexión significativa en su interacción con el mundo que les rodea. En este contexto, la tecnología ha emergido como un faro de esperanza, ofreciendo soluciones innovadoras para superar estas barreras. Uno de los desafíos más apremiantes en este campo es el desarrollo de sistemas de reconocimiento de señas en videos y la traducción precisa de lengua de señas a lenguaje natural. Este proyecto se adentra en este territorio, explorando formas de avanzar en la tecnología para hacer posible una comunicación fluida y sin barreras para las personas con discapacidad auditiva.

A lo largo de la historia, la tecnología ha desempeñado un papel crucial en la evolución de la comunicación, transformando la forma en que las personas se conectan y se entienden mutuamente. En el contexto específico de la comunicación para personas con discapacidad auditiva, hemos presenciado avances notables en el reconocimiento de señas y su traducción a lenguaje natural. Desde los primeros días de las máquinas de escribir de señas hasta los sistemas computarizados de reconocimiento de gestos, cada innovación ha allanado el camino para una comunicación más efectiva. La detección de lengua de señas en vídeos no es un problema nuevo en el área de visión por computadora, se lleva trabajando en él al menos dos décadas [2].

En el contexto específico de Venezuela, una nación cultural y lingüísticamente diversa, la Lengua de Señas Venezolana (LSV) ha evolucionado como una forma de comunicación distintiva y vital para la comunidad sorda. A medida de que la tecnología de reconocimiento de señas avanza en todo el mundo, existe una necesidad apremiante de adaptar estas

innovaciones a las características y particularidades de la LSV. Esta adaptación es esencial para garantizar la precisión y la efectividad del reconocimiento de la lengua de señas en situaciones específicas de Venezuela, incluyendo contextos educativos, sociales y profesionales.

Los avances en el reconocimiento de la lengua de señas han alcanzado un pico en las últimas décadas [3], sin embargo, a pesar de los avances significativos, los desafíos en el reconocimiento de señas en videos y la traducción de lengua de señas a lenguaje natural siguen siendo considerables, al punto donde incluso hoy día con la gran multitud de avances disponibles no se cuenta con sistemas accesibles y confiables para este propósito en todas las variantes nacionales y regionales de los lenguajes de señas. Los avances en el reconocimiento de señas se producen conforme se desarrollan nuevos trabajos en el área en cada país, y este trabajo busca ser un primer aporte a este ámbito en el contexto venezolano.

1.1 Antecedentes

A nivel global, se han construido motores de reconocimiento para aportar al campo del reconocimiento de la lengua de señas utilizando técnicas como el aprendizaje profundo (deep learning) y el procesamiento de imágenes. Estos trabajos han demostrado avances notables en la identificación y traducción precisa de gestos y movimientos que forman parte de la lengua de señas. Sin embargo, la mayoría de estos avances se han enfocado en lenguajes de señas más ampliamente estudiados, como el American Sign Language (ASL) y el British Sign Language (BSL). En el caso específico de la LSV, la investigación en esta área es aún limitada y se presenta como una oportunidad crucial para contribuir a la inclusión y el acceso a la información para las personas sordas en Venezuela.

Wu et al. [7] propuso una red neuronal convolucional de dos canales, con la singularidad de que a las imágenes se les aplicó preprocesamiento para detección de bordes antes de ingresarlas como entrada a la red neuronal, esta estrategia resultó en una precisión en la detección de gestos del 98.02%. Wadhawan et al. [8] propuso una arquitectura para redes neuronales convolucionales con el objetivo de reconocer gestos estáticos de lengua de señas, aplicada a la lengua de señas india, obtuvo una precisión de 98.85%, que usualmente sería un

resultado excelente, pero los videos que usó como entrada tenían un fondo blanco, lo que facilita la obtención de buenos resultados con esta técnica, y no representa una situación muy realista.

Saqib et al. [9] usó modelos de CNN para reconocer gestos estáticos y dinámicos en la lengua de señas pakistaní y consiguió una precisión de 90.79%. Neethu et al. [10] aplicó preprocesamiento consistente en segmentación a los dedos en las imágenes de su estudio, y esas imágenes fueron posteriormente clasificadas usando una CNN, todo esto para la detección de gestos, fue capaz de obtener una precisión de 96.2% en un dataset con fondos de colores simples, e iluminación variada.

María Blanco [11], como parte de su trabajo, construyó una recopilación de videos con gestos de la LSV usados en la Escuela de Ingeniería de Sistemas de la Universidad de los Andes, el cual consiste en un total de 134 señas, que incluyen palabras, y los números del 0 al 20 en LSV. Este conjunto no se tomó como el vocabulario objetivo para este trabajo debido a su extensión.

Amritanshu Kumar Singh, Vedant Arvind Kumbhare y K. Arthi utilizaron MediaPipe Holistic [36] para detectar y reconocer la pose humana en tiempo real [37]. Propusieron un marco que detecta la acción humana bajo diferentes condiciones y ángulos de visión que permiten la identificación de patrones divergentes basados en diferentes trayectorias espacio-temporales.

En el trabajo [38] se utilizó MediaPipe Holistic para el reconocimiento de la lengua de señas india. El estudio propuso un sistema robusto para el reconocimiento de la lengua de señas con el fin de convertir la lengua de señas india en texto o habla.

1.2 Justificación

Actualmente no existe igualdad de oportunidades en muchos ámbitos para los individuos sordos en Venezuela. No poder contar con un medio tan común para transmitir y recibir información como lo es la oralidad, representa una gran dificultad en la vida de

cualquier persona. Centrándonos más en el ámbito académico, los profesores que enseñan a alumnos con estas condiciones se enfrentan a dificultades adicionales para transmitirles sus conocimientos, puesto que, al no entender la lengua de señas, no es posible para el alumno comunicar sus dudas e ideas, salvo que cuente con la ayuda de un intérprete humano, que no siempre es posible. Los trabajos existentes centrados en aligerar o resolver esta problemática están enfocados a los lenguajes de señas de otros países, y en otros contextos. Este proyecto se justifica en la necesidad que existe por herramientas de este tipo en el contexto académico venezolano, más específicamente en la Escuela de Ingeniería de Sistemas de la Universidad de los Andes.

1.3 Objetivos

1.3.1 Objetivo General

- Desarrollar un motor de reconocimiento de la LSV, específicamente adaptado para el contexto de la Ingeniería de Sistemas.

1.3.2 Objetivos Específicos

- Recopilar una amplia variedad de videos y gestos que representen situaciones comunes en el contexto de la Ingeniería de Sistemas, asegurando la diversidad y autenticidad de los datos para el entrenamiento del modelo.
- Aplicar técnicas de preprocesamiento a los vídeos para facilitar la extracción de características de los mismos.
- Elegir y comparar arquitecturas de redes neuronales adecuadas para la detección y clasificación de gestos de la LSV.
- Construir un motor de reconocimiento mediante la aplicación de MediaPipe Holistic y redes neuronales profundas.
- Realizar pruebas exhaustivas del motor de reconocimiento utilizando métricas de precisión para verificar su utilidad práctica.

1.4 Alcance

Este proyecto se centrará en el desarrollo de un motor de reconocimiento de LSV, el cual recibe como entrada un flujo de video con gestos de LSV pertenecientes al contexto de la Ingeniería de Sistemas, e identificará los gestos que aparezcan en el flujo mediante tokens léxicos únicos para cada gesto. Este motor será la parte inicial de un sistema de traducción de LSV a lenguaje natural.

Se decidió enfocar el proyecto al conjunto de gestos usados en el contexto de la Ingeniería de Sistemas motivado en que existe una gran cantidad de gestos en la LSV, y las personas sordas de distintas profesiones suelen contar con incluso más gestos, por lo que un sistema de reconocimiento general de LSV resultaría complicado de hacer, pues implicaría dificultades adicionales en la recolección de vídeos para crear datasets de entrenamiento, y requeriría técnicas de entrenamiento bastante complejas para la red neuronal usada, así como varias fases de preprocesamiento de imágenes, debido a que, con una alta probabilidad, existirán gestos muy parecidos entre sí en ámbitos muy distintos. Un trabajo de esa escala incluso resultaría en la necesidad de hacer una clasificación más rigurosa de la LSV para poder realizarse satisfactoriamente, y eso escapa a los objetivos de este proyecto.

1.5 Metodología

La metodología elegida es Kanban. Según [40], “Kanban es una forma de ayudar a los equipos a encontrar un equilibrio entre el trabajo que necesitan hacer y la disponibilidad de cada miembro del equipo. La metodología Kanban se basa en una filosofía centrada en la mejora continua, donde las tareas se “extraen” de una lista de acciones pendientes en un flujo de trabajo constante.”

La aplicación de Kanban se consideró adecuada debido a que permite una gran visibilidad del progreso del trabajo de una forma sistemática, además de poseer la flexibilidad suficiente para adaptarse a los cambios continuos inherentes al desarrollo de cualquier software. Gracias a tener un indicador de progreso fácilmente cuantificable se pueden tomar decisiones con mayor grado de información al momento de establecer límites al trabajo realizado.

Capítulo 2

Marco Teórico

2.1 Introducción a la LSV

El establecimiento de la LSV se atribuye a José Arquero Urbano, quien fue una persona sorda de nacionalidad española que llegó a Venezuela en la década de 1950, lo hizo junto a Gustavo Alvarez Díaz y Eduardo López Ferrero, ellos fueron los fundadores de la Asociación de Sordomudos de Caracas, el objetivo de dicha institución fue velar por la plena participación de los sordos caraqueños en la sociedad, y parte de su labor también fue establecer un código claro para comunicarse entre ellos, que pasó a ser la LSV [4]. Ahora bien, la designación de LSV no fue inmediata, este término fue acuñado oficialmente a mediados de los ochenta, como se describe en [5].

La LSV representa una parte integral y distintiva de la cultura y la comunidad sorda de Venezuela. Es un sistema estructurado de comunicación gestual que permite a las personas sordas expresar pensamientos, emociones y conceptos sin depender del habla oral. Su importancia en la vida cotidiana no debe ser subestimada, ya que proporciona una forma rica y diversa de interacción y expresión. Esta forma única de comunicación ha evolucionado junto con la historia del país, adaptándose a las diversas regiones y comunidades dentro de Venezuela. Es un sistema lingüístico vivo que ha permitido a las personas sordas compartir sus historias, tradiciones y conocimientos de generación en generación.

Como ya se mencionó anteriormente, la creación de la LSV se atribuye a José Arquero Urbano. Se atribuye justamente por el hecho de que no hay investigaciones publicadas que hablen específicamente del origen de este lenguaje. Ciertamente existía una comunidad de personas sordas en Caracas antes de la llegada de Arquero, pero no existen evidencias documentadas de que estas personas usaran algún sistema de gestos específico antes de la creación de la LSV, así que tratar de remontarse a tal sistema sería únicamente especulación [13].

El estudio de la LSV no solo ofrece una ventana única para comprender la cultura y la experiencia de las personas sordas en Venezuela, sino que también plantea desafíos y oportunidades en el campo de la tecnología asistida. Con los recientes avances, se ha explorado el desarrollo de sistemas de reconocimiento de la lengua de señas que pueden facilitar la comunicación entre personas sordas y oyentes. Sin embargo, estos sistemas deben estar arraigados en una comprensión profunda de la LSV para ser efectivos.

La LSV es dinámica y se adapta constantemente para incorporar términos y expresiones emergentes. La incorporación de nuevos gestos se realiza por los practicantes de este lenguaje, para referenciar elementos que manejan frecuentemente en sus vidas cotidianas y en sus profesiones. Según [14], no existe un control riguroso sobre cuáles gestos se agregan oficialmente a la LSV, debido a la falta de una academia oficial de enseñanza de la misma, así que resultaría difícil documentar todas las señas que los habitantes sordos de Venezuela han agregado a su vocabulario, sin embargo, las variedades regionales de la LSV no resultan ininteligibles para sus usuarios [13].

2.1.1 Sintaxis de la LSV

Es fundamental reconocer que la LSV no es una forma simplificada de un idioma hablado; tiene su propia estructura gramatical y reglas lingüísticas distintas. La comprensión de estas características únicas es esencial para cualquier intento de desarrollar tecnologías de reconocimiento para este lenguaje. Según [13], los primeros trabajos descriptivos para la LSV, se publicaron en 1989, por investigadores de la Universidad de los Andes.

Las señas en la LSV tienen una gramática específica, lo que significa que el orden, la dirección y la forma de los movimientos tienen significados específicos. Esta complejidad gramatical es fundamental para la comprensión precisa del mensaje en el contexto de la LSV, y es la que aporta suficiente variabilidad para que existan suficientes gestos para que los practicantes de la LSV puedan referirse a cualquier palabra que necesiten en su vida cotidiana. Si bien muchos practicantes de la LSV han creado gestos para referirse a la jerga de su ambiente o profesión, otra opción con la que cuentan es la de usar el alfabeto dactilológico, el cual es un conjunto de gestos que representan las letras del abecedario, una a

una, así pueden describir términos que no tengan un gesto específico, sin necesidad de crear un gesto cada vez. En la figura 2.1, presente a continuación, se puede apreciar una representación de dicho abecedario, que fue publicado por FEVENSOR, en 1999.

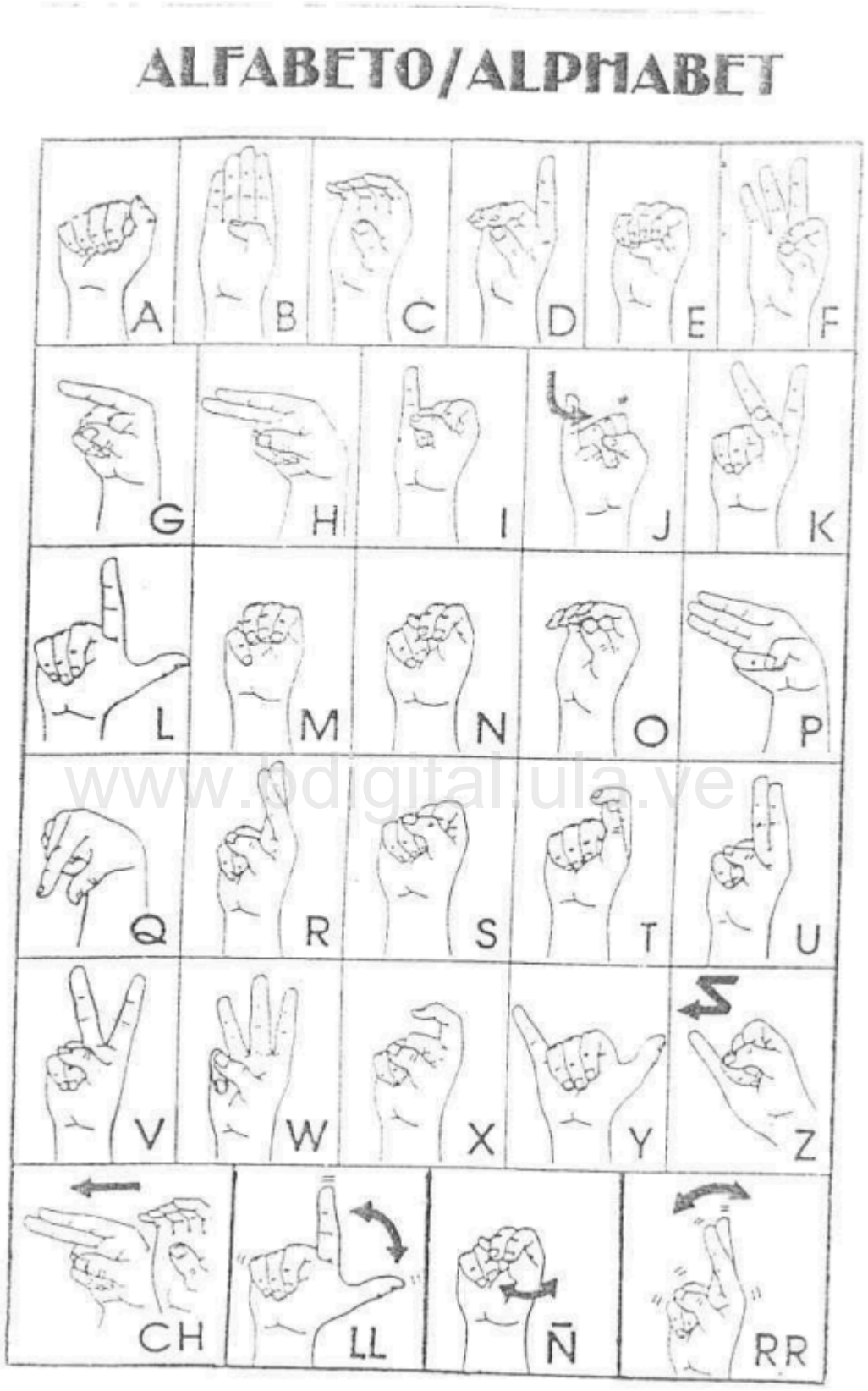


Figura 2.1 Abecedario dactilológico de la LSV. Obtenido de [15]

2.2 Tecnologías Actuales de Reconocimiento de Gestos

Las tecnologías para el reconocimiento de gestos han alcanzado un nivel de sofisticación sin precedentes, permitiendo una interacción intuitiva y natural entre humanos y computadoras. Estas tecnologías han evolucionado rápidamente, transformando la forma en que interactuamos con dispositivos electrónicos y sistemas informáticos. Según [16], el reconocimiento de gestos ha pasado de ser una tecnología emergente a convertirse en una herramienta muy importante en campos que van desde los videojuegos y la realidad virtual hasta la medicina y la industria.

Una de las tecnologías más destacadas en este campo es la utilización de sensores de profundidad, como el Kinect de Microsoft. Estos sensores emiten señales de luz que se reflejan en los objetos, permitiendo la captura precisa de movimientos tridimensionales. Investigaciones como [17] han demostrado cómo estos sensores pueden reconocer gestos complejos y sutiles, más específicamente, de la lengua de señas India, que goza de una complejidad inherente alta debido a que muchos de sus gestos se basan en superponer las manos, lo cual resulta desafiante al momento de reconocer los gestos, y su diccionario cuenta con más de 6000 palabras. Esta tecnología ha mostrado su valor en una variedad de campos, desde la rehabilitación física hasta la animación de personajes en tiempo real, y es una poderosa alternativa para su uso en el reconocimiento de señas.

Otra área de investigación prominente es el reconocimiento de gestos basado en visión por computadora. Según [18], la visión por computadora (CV por sus siglas en inglés) es un campo de las ciencias de la computación cuyo objetivo es dotar a las computadoras con la capacidad para identificar, ver y procesar información en una forma similar a las personas, y aprovechar dicha habilidad para resolver correctamente problemas que la requieran. La mayoría de algoritmos de CV se basan en la extracción de características (features) resaltantes en las imágenes, que pueden ser bordes de objetos, un color específico, una figura, o una combinación de todos los anteriores. Las técnicas tradicionales de CV requieren que se especifiquen los features a extraer de manera manual; una vez extraídos, se les aplican técnicas tradicionales de machine learning (ML) para su clasificación [19].

La realidad aumentada (AR por sus siglas en inglés) ha ampliado las posibilidades del reconocimiento de gestos al superponer información digital en el mundo real. Un ejemplo es el trabajo de [20], el cual se enfoca en el desarrollo de un modelo innovador de reconocimiento de gestos manuales basado en una Red Neuronal Convolucional Profunda (DCNN) para su implementación en entornos de AR. La investigación propone este modelo para controlar objetos virtuales en entornos tridimensionales utilizando gestos de mano. La tecnología usada permite detectar y rastrear los movimientos de la mano. Este estudio presenta un enfoque basado en aprendizaje profundo para mejorar la precisión y la eficiencia en el reconocimiento de gestos. El modelo se probó en dispositivos de AR/VR como HTC VIVE Pro y Kinect v2, demostrando su capacidad para controlar un manipulador industrial en tiempo real utilizando solo los movimientos de la mano.

Es apropiado mencionar, también, los sistemas de reconocimiento de gestos en tiempo real, como el trabajo de [2]. Este esquema de funcionamiento provee una gran practicidad al momento de aplicar en casos de uso cotidianos porque puede representar la base de la traducción de la lengua de señas en tiempo real, sin embargo, tales sistemas usualmente son más difíciles de entrenar con los métodos más convencionales por la complejidad de detectar los gestos en un flujo constante de vídeo, pero con la elección adecuada de herramientas se trata de un campo cuyos desafíos se pueden afrontar hoy día, y el presente trabajo es una prueba de ello.

2.3 Visión por computadora

La visión por computadora ha emergido como un campo interdisciplinario vital que fusiona la visión artificial y el aprendizaje automático, capacitando a las máquinas para interpretar y comprender el mundo visual que nos rodea. Este campo fascinante se ha vuelto esencial en diversas aplicaciones, desde reconocimiento facial y vehículos autónomos hasta diagnóstico médico y realidad aumentada. Según [21], la visión por computadora se centra en la extracción, análisis y comprensión de información visual (features) a partir de imágenes o vídeos, imitando así la capacidad humana de interpretar el mundo a través de la vista. Un feature se define como un parámetro descriptivo que se extrae de una imagen o video. Los features pueden usarse para interpretar contenido visual o como una medida de similitud para bases de datos de imágenes y vídeos [22].

Una de las áreas más emocionantes de la visión por computadora es el reconocimiento de objetos y patrones. Los sistemas de visión por computadora utilizan algoritmos para detectar y clasificar objetos en imágenes o videos. En fuentes como [22] se describen con rigurosidad dichos algoritmos, aunque es importante mencionar que en términos generales, los enfoques que combinan las técnicas de CV con técnicas de ML se han convertido en tendencia en las investigaciones, esto debido a que se suele lograr una buena precisión de esa forma. En la figura 2.2, presente a continuación, se puede apreciar una comparación del modus operandi del enfoque tradicional de visión por computadora, y del enfoque basado en redes neuronales, que se ha popularizado más recientemente.

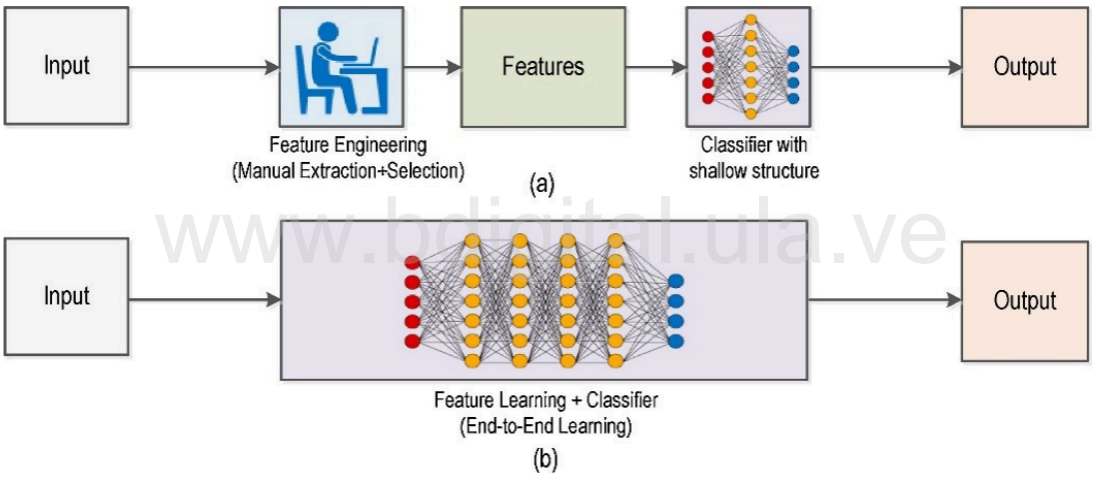


Figura 2.2 Comparación entre (a) técnicas clásicas de CV y (b) enfoques basados completamente en redes neuronales. Imagen obtenida de [25]

Otro aspecto significativo de la visión por computadora es la segmentación de imágenes. Los algoritmos de segmentación permiten dividir una imagen en regiones significativas o identificar objetos individuales dentro de una imagen. Estos métodos son esenciales en la industria médica para el análisis de imágenes de resonancia magnética y tomografías computarizadas, donde ayudan en la detección temprana de enfermedades y condiciones médicas. Además, la segmentación de imágenes es crucial en la industria del automóvil para el desarrollo de sistemas de asistencia al conductor y vehículos autónomos, donde permite detectar peatones, señales de tráfico y otros vehículos en tiempo real.

2.4 Machine Learning y Aprendizaje Profundo: La Base de las Redes Neuronales

En el vasto campo de la inteligencia artificial, Machine Learning (ML) representa una disciplina esencial. En su esencia, el ML permite a las computadoras aprender patrones y realizar tareas sin ser programadas explícitamente. Este aprendizaje se basa en el análisis de datos y la identificación de correlaciones, lo que habilita la toma de decisiones basada en datos de manera automatizada. El concepto de ML abarca una variedad de procedimientos estocásticos que pueden usarse para predecir el valor de un cierto parámetro basado en ejemplos similares a los que el algoritmo en cuestión haya sido expuesto. Entre los métodos básicos más conocidos de ML se incluyen técnicas como: Naïve Bayes, Random Forest, K-Nearest Neighbor, Logistic Regression, y Support Vector Machine [23]; estos métodos tienen en común que requieren una fase de entrenamiento, cuyo tipo varía dependiendo de la técnica elegida, y puede ser supervisada (con data etiquetada) o no supervisada (sin data etiquetada) antes de poder aplicar el método efectivamente. Si bien los métodos mencionados son más sencillos que aquellos que se utilizan en las ramas más especializadas de ML, presentan limitaciones cuando se enfrentan a tareas más complejas.

Recientemente, los enfoques básicos de aprendizaje automático han sido reemplazados en gran medida por arquitecturas más detalladas que emplean varias capas y pasan información en formato vectorial entre capas, refinando gradualmente la estimación hasta que se logra un reconocimiento positivo. Estos algoritmos suelen describirse como sistemas de "aprendizaje profundo" o redes neuronales profundas (DNN), y operan según principios similares a las estrategias de ML descritas anteriormente, aunque con mucha mayor complejidad. Según la estructura de la red, se utilizan comúnmente dos arquitecturas para una serie de tareas diferentes: redes neuronales recurrentes (RNN) que incluyen al menos una capa recurrente y redes neuronales convolucionales (CNN) que incluyen al menos una capa convolucional. Dependiendo del número y tipo de capas, estas redes pueden exhibir diferentes propiedades y generalmente son adecuadas para diferentes tipos de tareas, mientras que la fase de entrenamiento impacta decisivamente en el rendimiento del algoritmo [23].

Las redes neuronales convolucionales (CNN) son un tipo de red neuronal que puede extraer características de datos mediante estructuras convolucionales. A diferencia de los métodos tradicionales de extracción de características, las CNN no necesitan extraer características manualmente. La arquitectura de las CNN está inspirada en la percepción visual. Los kernel de las CNN representan diferentes receptores que pueden responder a diversas features; Las funciones de activación imitan la función existente en las neuronas biológicas de que sólo las señales eléctricas emitidas por las mismas que superen un determinado umbral puedan transmitirse a la siguiente neurona. Las funciones de pérdida y los optimizadores son mecanismos adicionales para tener un control más granular sobre qué aprende la CNN en cuestión [24]. Para facilitar la visualización de la estructura de una CNN tradicional, se incluye a continuación la figura 2.3:

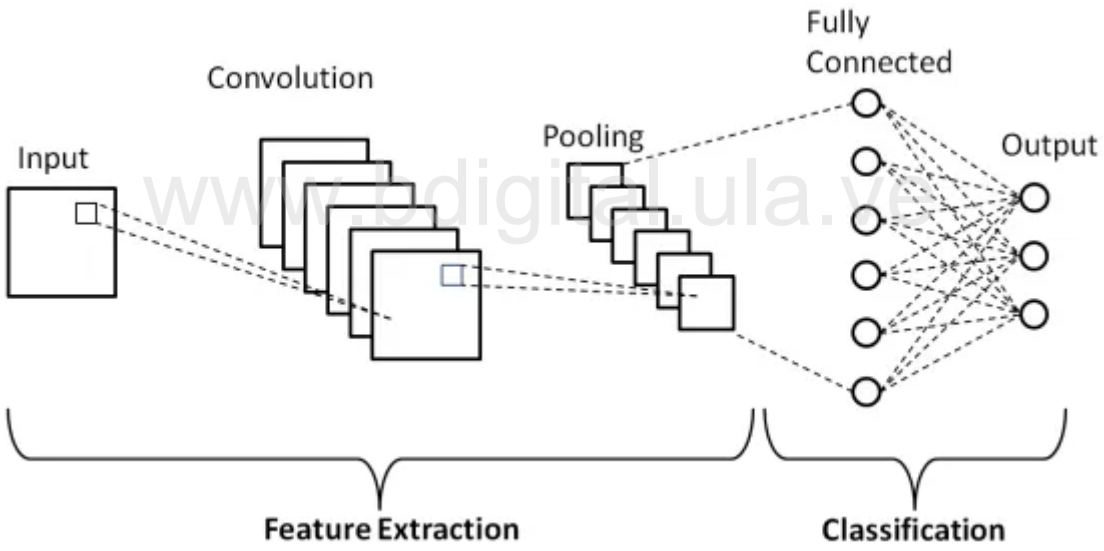


Figura 2.3: La estructura básica de una CNN. Imagen obtenida de [26].

Las aplicaciones de las CNNs se extienden más allá de la Visión por Computadora, abarcando áreas como el procesamiento del lenguaje natural, la medicina, la robótica y los vehículos autónomos. Estas redes han demostrado su capacidad para aprender y comprender patrones complejos en datos multidimensionales, lo que las convierte en una tecnología esencial para el futuro de la inteligencia artificial y la ciencia de datos.

2.5 Long Short-Term Memory: La herramienta enfocada en procesamiento secuencial

Antes de definir lo Long Short-Term Memory significa, es conveniente hablar de la super categoría que la contiene: Las redes neuronales recurrentes (RNN); son una clase de redes neuronales artificiales donde las conexiones entre nodos forman un grafo dirigido a lo largo de una secuencia temporal. Esto las hace aptas para procesar secuencias de datos de longitud variable, como series temporales o cadenas de texto [34]. Las RNN tienen "memoria" en el sentido de que la información puede persistir de un paso de tiempo al siguiente. Esto las hace útiles para tareas como el reconocimiento de voz o la traducción automática, donde la información relevante puede estar separada por intervalos de tiempo desconocidos [32]. Sin embargo, las RNN tradicionales sufren del problema del "desvanecimiento del gradiente", donde los errores de los pasos de tiempo lejanos se vuelven insignificantes durante el entrenamiento. Para superar este problema, se han desarrollado variantes de RNN, como las redes neuronales recurrentes de largo corto plazo (LSTM) y las redes neuronales recurrentes con compuertas (GRU) [35].

Long Short-Term Memory (LSTM) es un tipo de red neuronal recurrente (RNN) que ha demostrado ser muy eficaz en el modelado de secuencias y en el procesamiento de datos temporales. Propuesto por Hochreiter y Schmidhuber en 1997 [32], LSTM aborda el problema de la desaparición del gradiente en RNNs, lo que permite aprender dependencias a largo plazo en datos secuenciales.

La principal ventaja de LSTM radica en su capacidad para retener y utilizar información relevante durante largos períodos de tiempo, lo que lo hace especialmente adecuado para tareas que implican la comprensión de contextos complejos y la predicción a largo plazo. Esto se logra mediante la incorporación de unidades de memoria llamadas "celdas de memoria" que están diseñadas para almacenar y actualizar información a lo largo de varias iteraciones de tiempo. Las celdas de memoria en LSTM constan de tres compuertas principales: la compuerta de olvido, la compuerta de entrada y la compuerta de salida. Estas compuertas están diseñadas para regular el flujo de información dentro de la celda de memoria, permitiendo que se almacene, se olvide o se actualice la información según sea

necesario [33]. La compuerta de olvido controla qué información de la celda de memoria anterior se debe descartar, la compuerta de entrada determina qué nueva información se debe agregar a la celda de memoria y la compuerta de salida regula qué información se debe pasar a la salida de la celda.

El diseño de LSTM permite que la red aprenda a retener información relevante durante largos períodos de tiempo mientras filtra la información irrelevante o redundante. Esto lo hace especialmente útil en aplicaciones como el procesamiento de lenguaje natural (NLP), donde las dependencias a largo plazo son comunes y es crucial comprender el contexto completo de una secuencia de palabras.

Desde su introducción, LSTM ha sido objeto de numerosas extensiones y mejoras, incluidas variantes como LSTM bidireccionales, LSTM apiladas y LSTM con atención, que han demostrado ser aún más efectivas en una variedad de tareas de aprendizaje automático y procesamiento de secuencias [33]. En el contexto del reconocimiento de lengua de señas LSTM también se presenta como una poderosa alternativa para construir soluciones, ya que una de las características prevalentes en el lenguaje es la secuencialidad, esto aplica tanto para lengua de señas como para lenguajes orales, ya que el problema se centra en procesar una serie de imágenes o audio, donde cada segmento procesado no es completamente independiente de los anteriores, debido a esto este tipo de problema es un candidato válido para la aplicación de LSTM.

2.6 MediaPipe Holistic

MediaPipe Holistic es un marco de trabajo desarrollado por Google que permite la percepción simultánea de la pose humana, los hitos faciales y el seguimiento de las manos en tiempo real. Esta tecnología puede habilitar una variedad de aplicaciones importantes, como el análisis de fitness y deportes, el control mediante gestos, el reconocimiento de lengua de señas, los efectos de realidad aumentada, entre otros. MediaPipe Holistic consta de un pipeline con componentes de reconocimiento para pose, cara y mano optimizados que funcionan en tiempo real, con una transferencia mínima de memoria entre sus backends de inferencia. Cuando se incluyen todos los componentes, MediaPipe Holistic proporciona una topología unificada para un innovador conjunto de más de 540 puntos clave (33 de pose, 21

por mano y 468 hitos faciales) y logra un rendimiento cercano al tiempo real en dispositivos móviles [36].

Un aspecto interesante de MediaPipe Holistic es su capacidad para proporcionar una percepción holística y simultánea del lenguaje corporal, los gestos y las expresiones faciales. Esto es especialmente útil en aplicaciones como las interfaces de gestos remotos, la realidad aumentada de cuerpo completo, el análisis deportivo y el reconocimiento de la lengua de señas.

Vale la pena mencionar, que cada uno de los componentes de detección individuales de MediaPipe Holistic (Pose, manos y cara) son modelos individuales entrenados mediante técnicas de ML. Cada uno ha sido optimizado separadamente para su tarea específica, y al combinarlos todos se obtiene un marco de trabajo para detecciones de pose bastante poderoso. En la figura 2.4, a continuación, se puede apreciar una representación visual de algunos de los puntos clave que MediaPipe puede detectar en una imagen usando todos sus componentes:

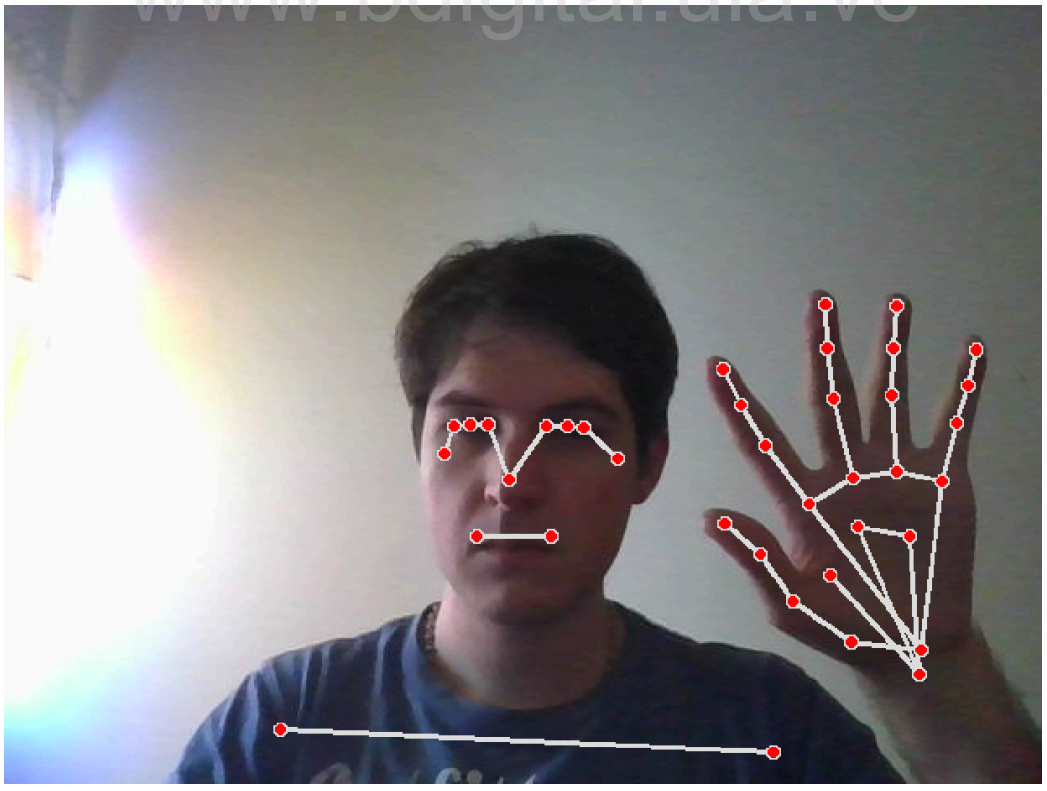


Figura 2.4: Representación visual de los puntos claves detectados usando MediaPipe Holistic.

Capítulo 3

Marco Metodológico

3.1 Diseño de la investigación

3.1.1 Tipo de investigación

Se trata de un estudio experimental centrado en el desarrollo y evaluación de un producto tecnológico específico, el motor de reconocimiento de la LSV. Según [27], los estudios experimentales buscan determinar si un conjunto de acciones específicas pueden influenciar las salidas observadas, en este caso el fenómeno a observar es la precisión del motor de reconocimiento.

El diseño experimental se eligió con base en que se especializa en la recopilación sistemática de datos cuantitativos, provenientes de la medición de las respuestas del motor en términos de precisión. El análisis estadístico de estos datos permitirá inferir conclusiones significativas sobre la efectividad del motor y respaldar las recomendaciones para mejoras futuras.

3.1.2 Enfoque de la investigación

El enfoque de este trabajo es cuantitativo, pues se busca medir y cuantificar la precisión del motor en la interpretación de gestos de la LSV. Según [27], el enfoque cuantitativo es un medio para probar teorías objetivo mediante la inspección de las relaciones entre las variables. Estas variables pueden ser medidas con instrumentos que nos den información numérica que pueda ser analizada usando procedimientos estadísticos.

La selección de este enfoque no fue únicamente por la recopilación de datos sino que también guió la elección de herramientas y métodos estadísticos para el análisis. El uso de métricas cuantitativas específicas permite la comparación sistemática de resultados y la inferencia de patrones significativos en el rendimiento del motor.

3.1.3 Identificación de la investigación de acuerdo al enfoque metodológico

La investigación siguió un enfoque metodológico positivista, buscando objetividad y generalización de resultados a través de la medición cuantitativa. Podemos definir el positivismo/postpositivismo como aquel que refleja una filosofía determinística acerca de las investigaciones donde las entradas posiblemente afectan las salidas. Por tanto, los problemas estudiados por los positivistas requieren la identificación de las causas que afectan dichas salidas, como se suele ver en los experimentos [27].

3.1.4 Variables

- Variable Independiente: Uso del motor de reconocimiento de la LSV.
- Variable Dependiente: Precisión en la interpretación de gestos de lengua de señas.

3.2 Población

La población en este estudio está compuesta por estudiantes de la carrera de Ingeniería de Sistemas en la Universidad de los Andes. Dada la naturaleza específica del motor de reconocimiento desarrollado, la población objetivo abarcó aquellos que pudieran verse beneficiados directamente de las tecnologías de asistencia relacionadas con la lengua de señas. La motivación de no usar una muestra comprendida únicamente por estudiantes que usen previamente la LSV es que al momento de realizar este trabajo no se encontraron suficientes estudiantes con esta habilidad previa para que la muestra fuese de un tamaño suficiente para el entrenamiento del modelo.

3.2.1 Muestra

La muestra específica para este estudio fue seleccionada de manera estratégica. Se realizó un muestreo intencional. La participación fue voluntaria, y se proporcionó información detallada sobre los objetivos y procedimientos del estudio antes de la inclusión en la muestra.

En el marco de esta investigación, se adoptó un enfoque de muestreo intencional para abordar la limitada presencia de individuos que utilizan la LSV como su principal medio de comunicación en la Facultad de Ingeniería de la Universidad de los Andes en Venezuela. Dada la escasez de este grupo específico, la muestra consistió principalmente de estudiantes de la misma facultad. Aunque estos estudiantes no utilizaran la LSV en su vida diaria, fueron capacitados para ejecutar los gestos de interés necesarios para el entrenamiento del modelo.

La elección de estos estudiantes como la base principal de la muestra se basó en la necesidad práctica de obtener videos representativos para el entrenamiento del modelo del motor de reconocimiento. La muestra fue instruida y orientada para realizar los gestos de manera precisa, independientemente de si emplean la LSV en su comunicación diaria.

3.2.2 Tipo de Muestra

La muestra se clasifica como una muestra no probabilística intencional debido a la naturaleza específica de la población objetivo y los criterios de inclusión. Dado que el interés principal es evaluar el rendimiento del motor de reconocimiento de la LSV en condiciones controladas, se seleccionaron participantes de manera deliberada con el fin de que la muestra sea representativa de los estudiantes que hacen vida en la facultad de ingeniería de la Universidad de los Andes. Priorizar este tipo de muestreo sobre uno general es uno de los factores que contribuye a que se requiera menor cantidad de datos para entrenar al modelo utilizado.

3.3 Instrumentos para la medición de resultados

En el marco de este estudio, la medición de resultados se llevó a cabo principalmente mediante instrumentos cuantitativos para evaluar de manera objetiva el rendimiento del motor de reconocimiento de la LSV. La selección de estos instrumentos se basó en la necesidad de recopilar datos precisos y cuantificables que permitan una evaluación rigurosa del sistema.

La evaluación cuantitativa se centró en la medición de una variable principal: la precisión de reconocimiento. La precisión se define como la fracción de los elementos de interés entre el total de elementos disponibles [28]. La expresión para calcularla es la siguiente:

$$Precisión = \frac{Número\ de\ gestos\ reconocidos\ correctamente}{Número\ total\ de\ gestos} \times 100$$

3.3.1 Validación de los instrumentos

La validación rigurosa de los instrumentos cuantitativos utilizados para medir la precisión de reconocimiento en el motor es esencial para garantizar la fiabilidad y validez de los resultados obtenidos. En este contexto, la validación se concibe como un proceso crucial que busca confirmar la idoneidad y precisión de los instrumentos de medición [29].

La validación se llevó a cabo mediante una prueba piloto, involucrando a un participante representativo de la población objetivo. Durante esta prueba, se evaluó la consistencia de las mediciones obtenidas a través de los instrumentos cuantitativos. La retroalimentación del participante proporcionó percepciones valiosas acerca de qué detalles es posible mejorar en lo que respecta a la recolección de datos pertinentes para el entrenamiento del motor.

Además, se aplicó el análisis de confiabilidad para evaluar la consistencia interna de los instrumentos cuantitativos [30]. Este análisis permitió determinar si los instrumentos midieron de manera coherente la variable de interés, en este caso, la precisión de

reconocimiento. La confiabilidad de los instrumentos es esencial para obtener resultados consistentes y replicables, fundamentales en la validez de la evaluación del rendimiento del motor.

3.4 Procesamiento de datos

En la fase de procesamiento de datos, fue necesario aplicar un conjunto de técnicas necesarias para preparar los datos usados en el entrenamiento del motor. El primer paso fue llevar a cabo una revisión exhaustiva para garantizar la integridad y consistencia de los conjuntos de datos recopilados. Esto implica la identificación y manejo de posibles valores atípicos o errores que pudieran surgir durante la recopilación, asegurando así la calidad de los datos analizados. La estrategia de manejo elegida para los datos atípicos fue el descarte de los mismos, puesto que luego de inspeccionar los datos recolectados, se encontró que la cantidad de datos atípicos presentes eran despreciables respecto al total de datos recolectados.

El dataset de videos completo sigue una estructura consistente en carpetas identificadas por la palabra asociada al gesto que aparece en los videos que están dentro de la misma, por ejemplo, los videos del gesto “ecuación” se encuentran dentro de la carpeta “ecuación”, el primer video con el nombre “0.mp4”, el segundo “1.mp4”, y así sucesivamente. Los videos del gesto “ecuación” de otros sujetos también se ubicaron bajo la carpeta mencionada anteriormente, pero se enumeraron a partir de un número mayor que la última secuencia del sujeto anterior.

A continuación, se recorrió cada carpeta del dataset, así como cada vídeo presente dentro del mismo. Los videos tienen una duración aproximada de un segundo, por limitaciones en el software de captura no fue posible que la longitud de los videos fuera completamente uniforme en todos los casos. En cada vídeo se procesaron los primeros 29 frames (cantidad determinada por el video con menor duración de todo el dataset), aplicando el reconocimiento de MediaPipe, lo cual permitió obtener un listado de puntos claves que detallan las posiciones X, Y, y Z de cada uno, por conveniencia, este listado se transformó a un vector unidimensional compuesto por las coordenadas de cada punto clave detectado, en sucesión. Para el caso donde MediaPipe no pudiera identificar algún punto clave en la imagen (por ejemplo cuando las manos no aparecen en la misma) se manejó agregando tantos ceros

como fuese necesario, lo cual fue posible porque la cantidad de puntos clave detectables por MediaPipe es constante para cada parte del cuerpo. Este proceso resultó en la obtención de un vector con 1662 valores numéricos por cada frame de video, cada uno de estos vectores, fue persistido al sistema de archivos en forma de un arreglo de numpy, bajo una estructura de carpetas similar a la del dataset recolectado, con la diferencia, que en el nivel de la jerarquía de carpetas donde estaban los videos numerados, se encuentran carpetas con la misma numeración en su lugar, y dentro de cada una de ellas, hay 29 archivos, los cuales son los arreglos de numpy resultantes de detectar los puntos clave en cada uno de los primeros 29 frames del video en cuestión.

El formato de arreglos de numpy posee varias ventajas para la tarea de entrenar una red neuronal, siendo la primera de estas lo ligero del formato, que ya no es un archivo de video, sino una serie de valores numéricos, que ocupan menos espacio de almacenamiento, y dada la cantidad de datos con los que se trabajó, hizo posible su carga en memoria, lo que facilitó el proceso de entrenamiento del modelo.

Capítulo 4

Implementación y pruebas

4.1 Diseño y Arquitectura del Sistema

El lenguaje elegido para la implementación del motor fue Python, la elección del mismo como lenguaje de programación para implementar el motor de reconocimiento de la LSV se fundamentó en su simplicidad, flexibilidad y el amplio soporte para bibliotecas de ciencia de datos y aprendizaje profundo [41]. Python es particularmente valorado en la comunidad de inteligencia artificial por su sintaxis intuitiva y por facilitar la experimentación rápida, características esenciales para la investigación y desarrollo en campos innovadores como el reconocimiento de gestos. Además, bibliotecas como TensorFlow y PyTorch ofrecen capacidades avanzadas para la implementación de redes neuronales, lo que hizo de Python una opción robusta y versátil para este proyecto.

Para implementar el motor, se eligió una arquitectura basada en bucle, también conocida como ciclo de control, que emerge como una solución robusta y eficiente para enfrentar las demandas propias de un sistema que funciona en tiempo real. Este enfoque se caracteriza por su capacidad para ejecutar un conjunto de operaciones de manera secuencial y repetitiva, adaptándose continuamente a los cambios en los datos de entrada. En este contexto, la elección de una arquitectura basada en bucle para el motor de reconocimiento de señas se justificó por varios factores clave.

Primero, la naturaleza dinámica del reconocimiento de señas, donde los gestos y las expresiones cambian rápidamente y varían en complejidad, requiere un sistema que pueda procesar flujos de video de manera continua sin interrupciones. La arquitectura basada en bucle facilita esto al mantener el sistema en un estado de operación constante, procesando secuencialmente cada frame del video a medida que llega [42]. Este modelo es adecuado para aplicaciones en tiempo real, donde la capacidad de respuesta y la adaptabilidad son críticas.

Además, la arquitectura basada en bucle permite la integración efectiva de mecanismos de retroalimentación, una ventaja significativa para el procesamiento de señas. Estos mecanismos pueden ajustar dinámicamente los parámetros del sistema o refinar los algoritmos de reconocimiento en función de los resultados obtenidos, mejorando así la precisión y la eficiencia del motor [43]. Por ejemplo, si el sistema identifica incorrectamente un gesto, puede ajustar sus parámetros para mejorar el reconocimiento en iteraciones futuras, una capacidad esencial para un aprendizaje continuo y la optimización del rendimiento.

En el desarrollo de tecnologías para la interpretación de la lengua de señas, la eficiencia en el procesamiento de datos es fundamental. La arquitectura basada en bucle optimiza el uso de recursos al enfocarse exclusivamente en los datos relevantes en cada iteración del ciclo, evitando el procesamiento innecesario de información irrelevante. Esto se traduce en una mayor velocidad de procesamiento y en una reducción del tiempo de respuesta, aspectos esenciales para asegurar una interpretación fluida y precisa de las señas en tiempo real.

Las acciones del sistema en cada iteración se conciben como cuatro fases principales: la fase de captura, la fase de preprocesamiento, la fase de reconocimiento y la fase de presentación de resultados. A continuación, en la figura 4.1, se puede apreciar un diagrama sencillo que muestra dichas fases:

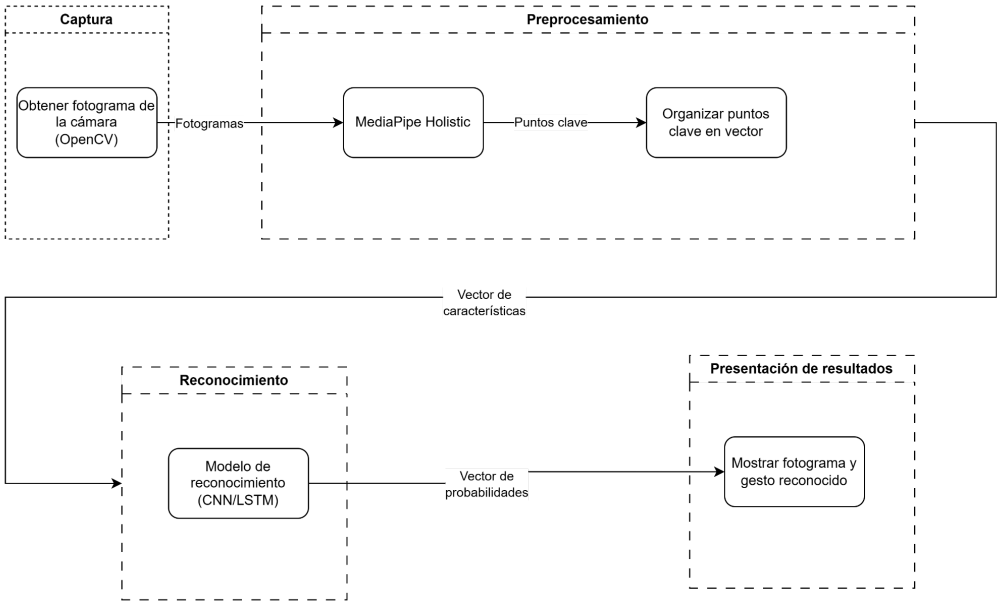


Figura 4.1 Diagrama de las acciones del sistema en funcionamiento

Para la fase de captura, se emplea la entrada indicada al motor de reconocimiento, que puede ser cualquier tipo de entrada soportada por OpenCV-Python, que es el paquete de Python que funciona de interfaz con la popular biblioteca OpenCV, esto incluye las cámaras conectadas al equipo; así como archivos de video o transmisiones de video en línea. La elección de OpenCV como biblioteca para manejar la captura de imágenes se basó en su amplio conjunto de herramientas optimizadas para el procesamiento de imágenes y visión por computadora. Bradski [44] introdujo OpenCV como una biblioteca de software libre que proporciona infraestructura y herramientas necesarias para el procesamiento de imágenes en tiempo real. Su soporte para múltiples plataformas y lenguajes de programación, junto con una comunidad activa y una extensa documentación, hacen de OpenCV una opción robusta y versátil para el desarrollo de sistemas avanzados de visión por computadora. Gracias a la funcionalidad de OpenCV, resultó sencillo acceder a la cámara del equipo donde se ejecutó el software, lo que permitió ejecutar las fases siguientes para cada fotograma capturado.

En la fase de preprocesamiento, se usa MediaPipe Holistic, que permite el seguimiento simultáneo de la pose del cuerpo, las manos y los hitos faciales en tiempo real. Esta tecnología se aplicó a los fotogramas capturados de videos para identificar y etiquetar los puntos clave de la lengua de señas antes de su análisis por las redes neuronales. La elección de MediaPipe Holistic para esta etapa se debió tanto a su capacidad para proporcionar una percepción holística del lenguaje corporal, crucial para la interpretación precisa de las señas, como a su resistencia a elementos de la imagen que suelen representar un desafío en muchas tareas de visión por computadora, como la iluminación, los elementos de fondo, el color de la ropa del sujeto, entre otros. Al obtener el conjunto de puntos clave expresados en coordenadas gracias a MediaPipe, éstos se organizaron y concatenaron en un único vector que sirvió de entrada a la fase de reconocimiento. Dicho vector posee 1662 elementos, para explicar esta cifra, se consideran 33 puntos clave de pose del cuerpo donde cada uno tiene atributos x, y, z y un valor de visibilidad que va de 0 a 1 (para un total de 132 elementos); 21 puntos clave por cada mano con 3 atributos cada uno, ya que solo los puntos claves de la pose de cuerpo incluyen visibilidad (esto resulta en 126 elementos); y 468 hitos faciales con 3 atributos cada uno (1404 elementos). La suma de estos componentes da un total de 1662 elementos. En algunos casos, MediaPipe no fue capaz de detectar todos los puntos clave en la imagen proveniente de la cámara, principalmente debido a que puede que solo una parte del cuerpo de la persona se encontrara enfocada en ese momento, la estrategia adoptada en

dichos casos fue reemplazar con ceros aquellos puntos clave que no se detecten, manteniendo las dimensiones del vector mencionado previamente.

La fase de reconocimiento consiste en la aplicación del modelo elegido para identificar el gesto en la secuencia de fotogramas capturados. Se exploraron dos arquitecturas de red neuronal distintas: CNN y LSTM. En este punto del proceso, como es de esperarse, se le suministró el vector obtenido en la fase de captura a la red neuronal de elección, para obtener su salida correspondiente: un vector de probabilidades, donde cada una está asociada a una clase (palabra) conocida por el modelo, la suma de todas las probabilidades expresadas en este vector es 1. Se siguieron algunos criterios descritos a detalle en la sección 4.2 de este trabajo para determinar una palabra de este vector como reconocida. Esta fase evaluó ambos tipos de arquitecturas propuestas en términos de precisión, y pérdida (loss) para determinar cuál ofrece un mejor rendimiento para el reconocimiento de señas específicas de la LSV.

Finalmente, en la fase de presentación de resultados, el motor se encarga de escribir a su salida el flujo de gestos reconocidos, esto se logra mediante la aplicación de un callback, que es uno de los parámetros que el motor requiere para funcionar; el mismo se ejecuta cada vez que un gesto es reconocido, y recibe como argumento una cadena de caracteres identificando el gesto en cuestión, de esta manera, el usuario es capaz de especificar cualquier medio de salida que desee, y esto le da la máxima flexibilidad posible al motor, la motivación de este diseño fue facilitar el posterior desarrollo de un sistema de traducción de LSV a lenguaje natural, ya que dicho sistema, pudiera integrarse fácilmente con el motor desarrollado en este trabajo. Otro de los parámetros que el motor puede recibir, es una bandera que indica si se desea ejecutar en modo depuración, en este modo, al ejecutarse el motor presenta al usuario los fotogramas capturados en forma de vídeo, tal como una aplicación de cámara, pero sobre dicha visualización se muestran 2 elementos de interés: el primero ubicado en la parte superior de la pantalla, se reserva un pequeño espacio para mostrar los gestos que son identificados por el motor, en forma de un flujo de gestos; el segundo se encuentra del lado izquierdo de la interfaz, y se trata de una lista vertical con los gestos soportados por el motor actualmente, solapado con cada palabra hay un indicador de seguridad en el reconocimiento, que representa la probabilidad que asigna el modelo al gesto asociado, con el programa en funcionamiento, se puede observar cómo estos indicadores varían constantemente, producto de que se procesan fotogramas continuamente en el motor, esta representación visual es valiosa porque a diferencia de solo mostrar el gesto reconocido,

se puede ver qué otros gestos tienen probabilidades significativas durante el reconocimiento, esto es especialmente importante en caso que un gesto no se identifique consistentemente de forma correcta, ya que al inspeccionar las probabilidades en tiempo real, es posible identificar motivos para la falla, para ilustrar esto último, si en el proceso de identificación se produce un resultado incorrecto, y sólo crece el indicador del gesto incorrecto, eso podría indicar que las predicciones del motor favorecen esa clase (gesto), muy posiblemente por falta de calidad en las muestras de entrenamiento; otro escenario sería si aumenta tanto el indicador del gesto correcto como de uno incorrecto, esto indicaría que al motor le cuesta distinguir estas dos clases, y sería recomendable incluir más muestras de las mismas durante el entrenamiento.

Es importante mencionar que los fotogramas capturados por el software no se procesan como un flujo único de video en la fase de reconocimiento, sino que se agrupan en lotes de 29 fotogramas cada uno, por lo que si bien se preprocesan todos los fotogramas captados por la cámara, para realizar el reconocimiento, se espera a acumular la cantidad mencionada previamente, y dicho lote de fotogramas sirven de entrada al modelo para efectuar el reconocimiento. Con la cámara usada para el desarrollo del motor, estos 29 fotogramas equivalen a un lapso muy cercano a un segundo, que representa la frecuencia con la que se efectúa la fase de reconocimiento, este intervalo de tiempo se eligió por practicidad mientras se desarrollaba el software.

Como detalle adicional respecto a la fase de presentación de resultados, se incorporaron una serie de criterios para filtrar los resultados de reconocimiento mostrados al usuario a modo de que los mismos fuesen más fácilmente interpretables: el primer criterio es que para un gesto sea aceptado como reconocido, la seguridad del mismo debe ser mayor a un umbral de seguridad, que es uno de los parámetros que recibe el motor, en caso de no suministrarse tiene un valor por defecto de 0.50 (50%), esto se prueba en el vector de probabilidades producto de la fase de reconocimiento; el segundo criterio es que un mismo gesto no se mostrará como reconocido más de una vez consecutiva en la aplicación, si bien el modelo puede reconocer el mismo gesto varias veces seguidas, se eligió mostrar únicamente una, hasta que se reconozca otro gesto distinto, esto puede sonar inconveniente, pero resultó en mejoras importantes de estabilidad durante las pruebas prácticas del motor, por lo que se incluyó como parte de la fase de presentación de resultados; el siguiente criterio está ligado al anterior, y consiste en que un gesto debe ser la ocurrencia más común entre las últimas 10 salidas del modelo antes de mostrarse como reconocido para el usuario, debido a que si un

mismo gesto se reconoce varias veces en un lapso corto, es mucho más probable que sea el gesto que aparece en la entrada; el criterio final aplicado es el de evaluar los puntos clave pertenecientes a las manos que provengan de la entrada, para compararlos con los de fotogramas anteriores, con el objetivo de determinar si las manos del usuario se reconocen en la entrada, y qué tanto se han movido de un fotograma al otro, de esta manera, se puede evitar que el motor emita resultados tanto en el escenario que las manos del usuario no estén presentes en la entrada (lo cual imposibilita la ejecución de señas de la LSV) como que la mismas no se estén moviendo en absoluto, que usualmente corresponde a cuando el usuario no está ejecutando ningún gesto.

Antes de incorporar los criterios mencionados, el proceso de mostrar los resultados al usuario ocurría muy rápido para que el mismo fuera útil: el motor daba muchas salidas inesperadas entre los resultados correctos, anulando su utilidad práctica.

4.2 Desarrollo del Motor de Reconocimiento

El desarrollo del motor se realizó utilizando el ambiente de desarrollo Jupyter Notebook, el cual es un proyecto de código abierto sin fines de lucro, surgido del Proyecto IPython en 2014, que evolucionó para admitir la ciencia de datos interactiva y la informática científica en todos los lenguajes de programación. Jupyter es un software 100% de código abierto, gratuito para utilización bajo los términos liberales de la licencia BSD modificada [11]. Esta clase de entorno es muy beneficioso en los proyectos de modelos de ML de todo tipo, por las ventajas inherentes que ofrece segmentar el código, con facilidades de documentación que normalmente no son posibles en un editor de texto tradicional, y con el añadido de que permite desarrollar el código de una forma iterativa sin incrementar su complejidad con abstracciones adicionales.

Para la recolección de datos se empleará la cámara de un celular Techno Spark 20, los videos tomados tendrán dimensiones de 2560 x 1440 píxeles. Ya que se tomará una cantidad importante de clips de vídeo, se desarrolló una aplicación sencilla para este fin, la misma se encarga de guardar los videos capturados con nombres numéricos secuenciales bajo una carpeta que identifique el gesto al que pertenece el video, la motivación para esto fue lograr que el proceso de captura de datos fuese más sistemático, eliminando la necesidad de

clasificar cada clip de video manualmente de acuerdo a su gesto correspondiente. La aplicación se creó en forma de un script ejecutable mediante la aplicación DroidScript [39]. En la figura 4.2, presente a continuación se puede ver la interfaz de la aplicación usada para capturar los datos:

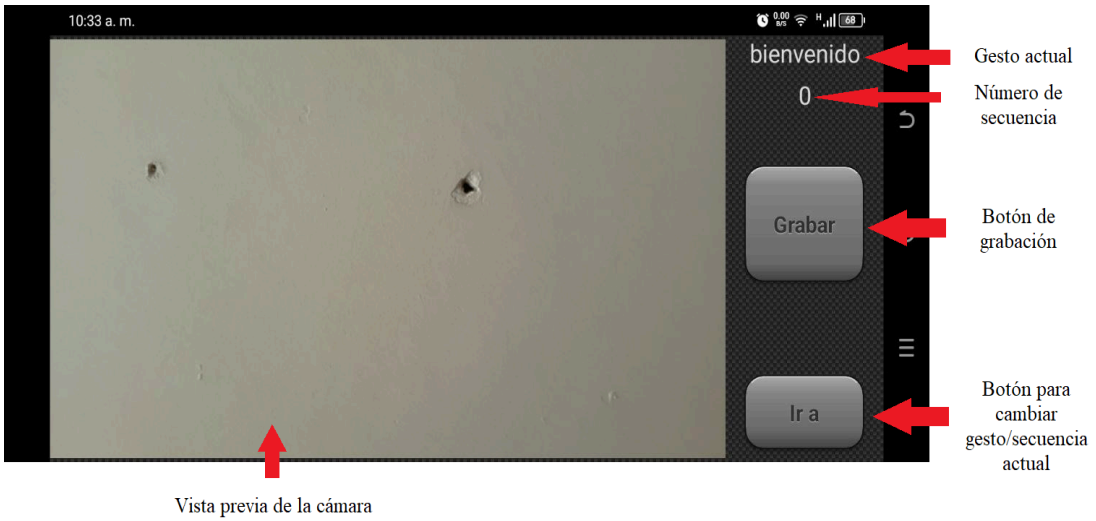


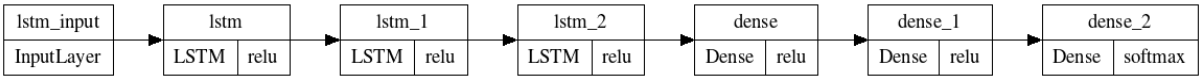
Figura 4.2 Interfaz con anotaciones ilustrativas de la aplicación de captura de datos

4.3 Entrenamiento

La evaluación del motor de reconocimiento de la LSV se centró en dos fases fundamentales: el rendimiento durante el entrenamiento del modelo y el rendimiento durante su ejecución en la práctica. Esta dualidad en la evaluación es esencial para comprender tanto la capacidad del modelo para aprender de los datos proporcionados como su efectividad en situaciones del mundo real, donde la variabilidad y la imprevisibilidad de las señas presentan desafíos adicionales.

El entrenamiento del modelo constituyó la fase inicial crítica, donde se aplicaron técnicas de aprendizaje profundo para ajustar los parámetros del modelo basado en el conjunto de datos recolectado. La efectividad del entrenamiento se midió mediante 2 métricas estándar: la precisión (accuracy), y el valor de la función de pérdida (loss), luego de cada época de entrenamiento.

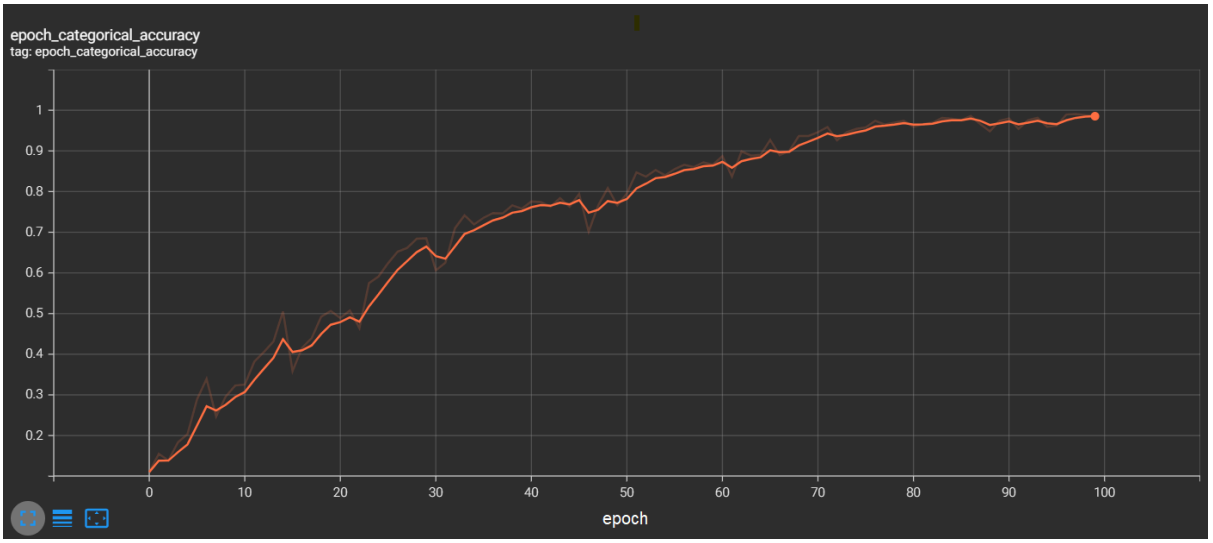
Durante esta fase, para las dos arquitecturas evaluadas se observó una mejora progresiva en la precisión y una disminución en la pérdida a medida que incrementó el número de épocas de entrenamiento, lo que indica que ambas arquitecturas, aunque simples, poseen el nivel de complejidad suficiente para adaptarse apropiadamente al conjunto de datos recolectados. A continuación en la figura 4.3, se presenta el esquema del modelo LSTM entrenado así como el resumen y resultados durante el entrenamiento:



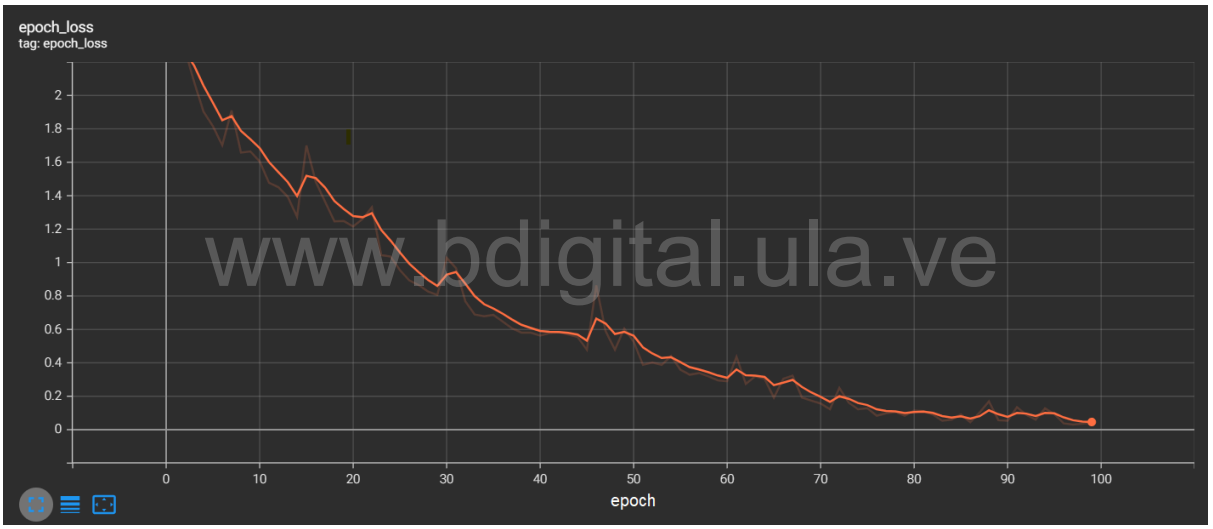
a) Resumen de la arquitectura del modelo

```
Model: "sequential_6"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_16 (LSTM)              (None, 29, 128)           916992

 lstm_17 (LSTM)              (None, 29, 256)           394240

 lstm_18 (LSTM)              (None, 128)               197120

 dense_15 (Dense)            (None, 256)               33024

 dense_16 (Dense)            (None, 32)                8224

 dense_17 (Dense)            (None, 10)                330

=================================================================
Total params: 1549930 (5.91 MB)
Trainable params: 1549930 (5.91 MB)
Non-trainable params: 0 (0.00 Byte)
```

b) Diseño del modelo



c) Precisión vs número de época durante el entrenamiento



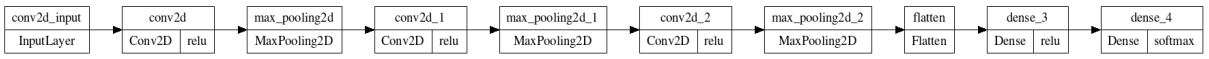
d) Pérdida vs número de época durante el entrenamiento

Figura 4.3 Resumen del modelo LSTM entrenado

El entrenamiento para el modelo LSTM consistió en 100 épocas, para generar las gráficas de precisión y pérdida durante el entrenamiento se usó TensorBoard, que es una herramienta presentada como un kit de visualización para TensorFlow [45], ofrece la posibilidad de graficar el progreso del entrenamiento mientras este se realiza, mediante la inspección de los archivos de registro (Logs) que TensorFlow guarda luego de cada época del entrenamiento, la herramienta también provee la posibilidad de suavizar las curvas de las métricas tomadas, que se aplicó con todas las figuras del presente trabajo para una mejor

visualización de los resultados. Luego de las 100 épocas el modelo logró una precisión del 97.43% en los datos de prueba, y el valor alcanzado por la función de pérdida fue de 0.0347

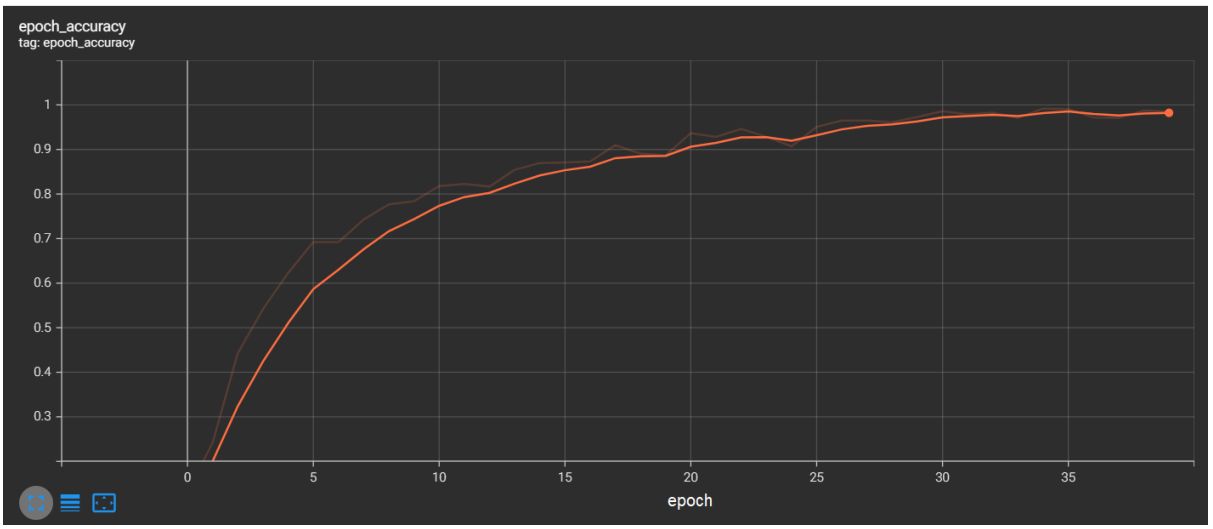
Para dar continuidad, en la figura 4.4, presentada en breve, se encuentra un resumen similar para el modelo CNN entrenado:



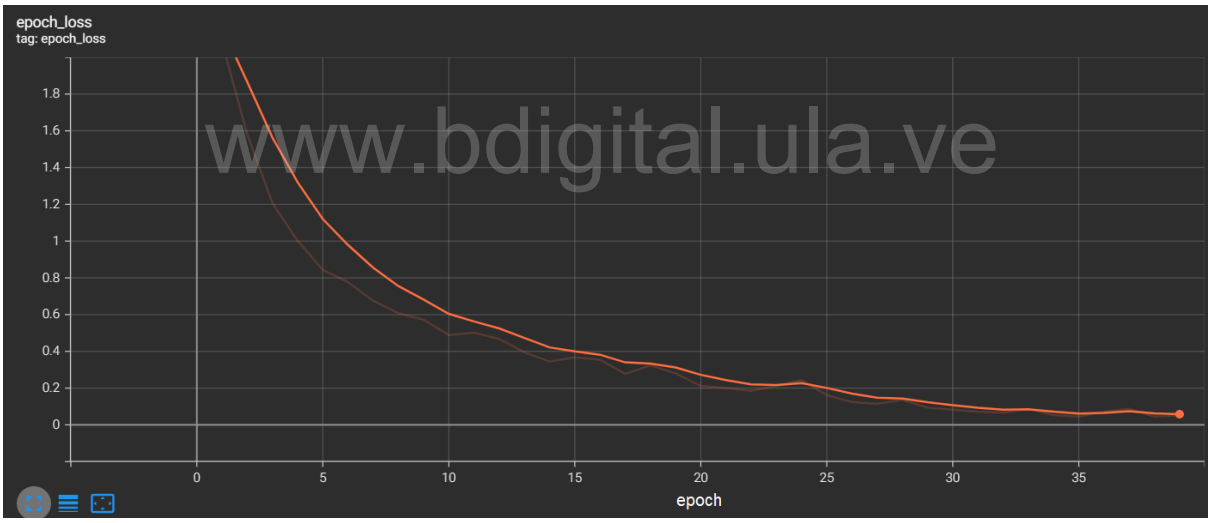
a) Resumen de la arquitectura del modelo

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_6 (Conv2D)           (None, 27, 1660, 32)      320

 max_pooling2d_6 (MaxPoolin  (None, 13, 830, 32)       0
 g2D)

 conv2d_7 (Conv2D)           (None, 11, 828, 64)       18496

 max_pooling2d_7 (MaxPoolin  (None, 5, 414, 64)        0
 g2D)

 conv2d_8 (Conv2D)           (None, 3, 412, 32)        18464

 max_pooling2d_8 (MaxPoolin  (None, 1, 206, 32)        0
 g2D)

 flatten_2 (Flatten)         (None, 6592)              0

 dense_4 (Dense)             (None, 32)                210976

 dense_5 (Dense)             (None, 10)                330

=================================================================
Total params: 248586 (971.04 KB)
Trainable params: 248586 (971.04 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

b) Diseño del modelo



c) Precisión durante el entrenamiento



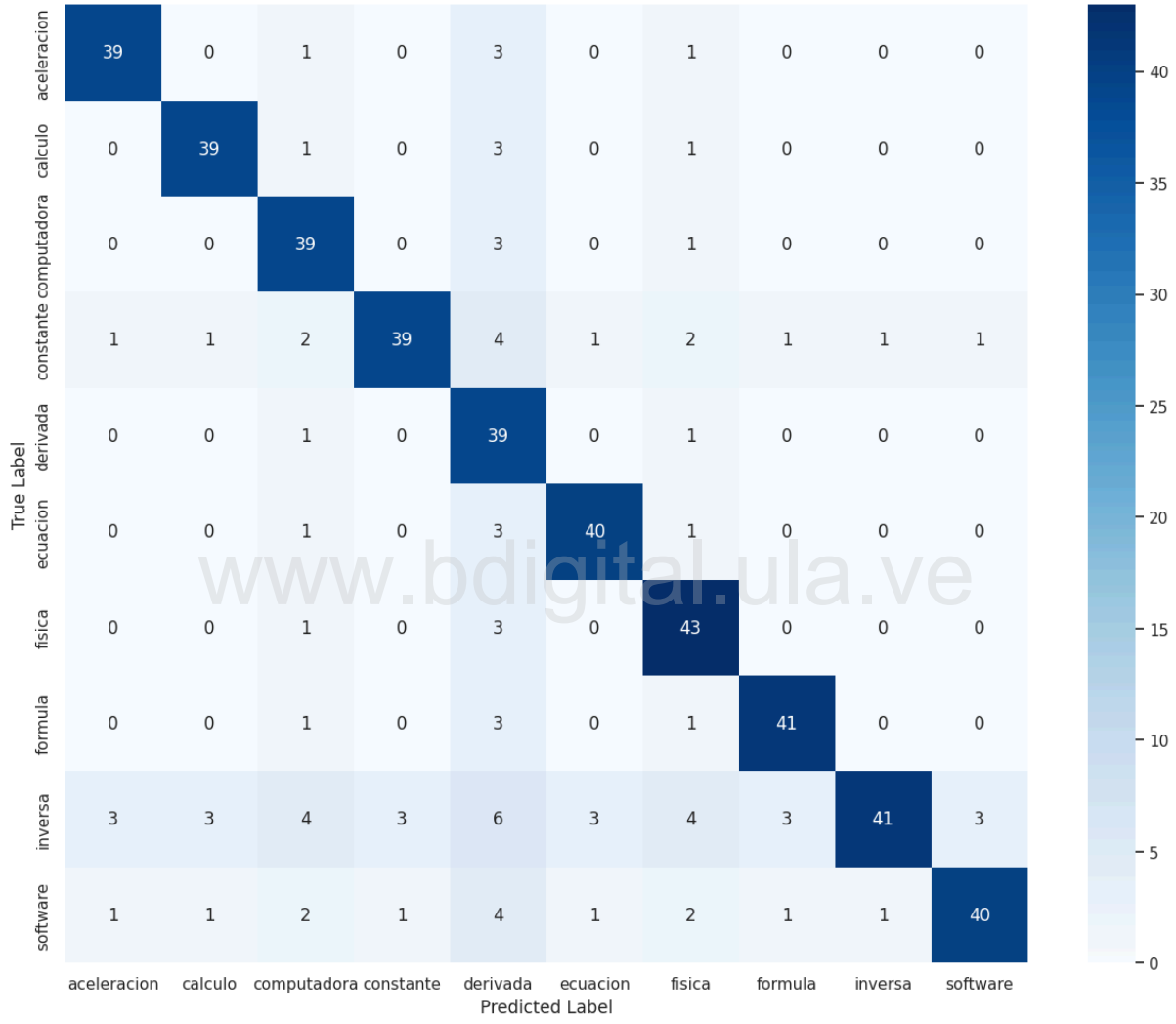
d) Pérdida durante el entrenamiento

Figura 4.4 Resumen del modelo CNN entrenado

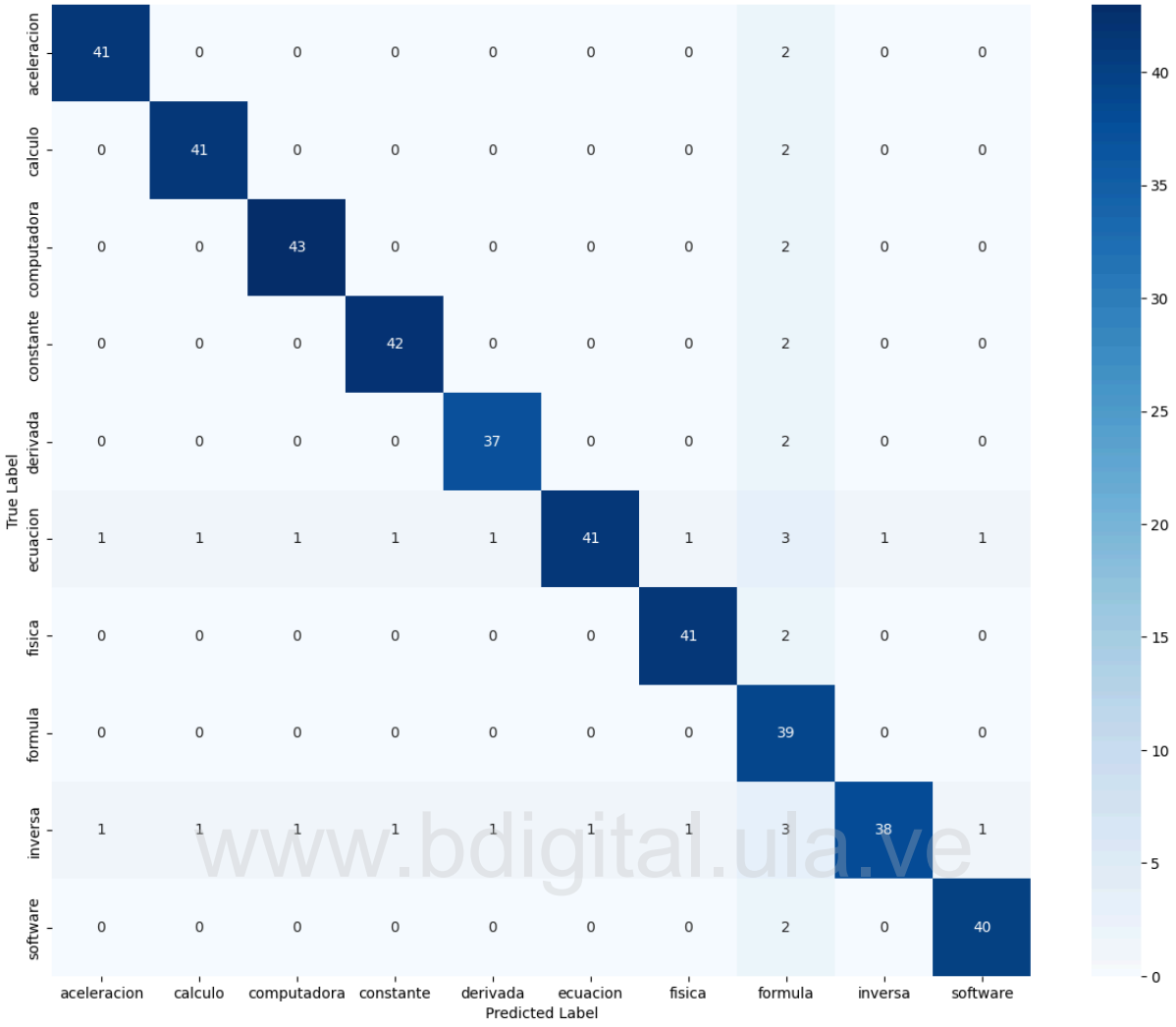
El entrenamiento para el modelo CNN consistió en 40 épocas, que es la cantidad que se determinó adecuada al inspeccionar el comportamiento de la precisión y la pérdida durante el desarrollo del modelo, esto último es evidente en la figura 4.4, presentada anteriormente. Vale la pena mencionar que aún con una menor cantidad de épocas, el tiempo de entrenamiento total para esta arquitectura fue comparable al del modelo LSTM, ya que el tiempo requerido

por época resultó mayor. Luego de las 40 épocas el modelo logró una precisión del 98.47% en los datos de prueba, y el valor alcanzado por la función de pérdida fue de 0.0514

A continuación, en la figura 4.5, se incluyen las matrices de confusión para ambos modelos:



a) LSTM



b) CNN

Figura 4.5 Matrices de confusión de los modelos entrenados

4.4 Desafíos y Soluciones

El desafío principal que se enfrentó al desarrollar el proyecto fue en la fase inicial del mismo. Originalmente se planteó el uso de CNN con un enfoque tradicional: que los fotogramas de vídeo pasaran por una fase de preprocesamiento estándar (conversión a escala de grises seguida de reducción de la resolución y aumento del contraste), para luego servir de entrada directamente a la CNN (puesto que una imagen en escala de grises puede representarse como un arreglo bidimensional donde cada posición del mismo representa la intensidad de color de un pixel, con 0 siendo negro, y 255 siendo blanco). El problema con

dicho enfoque radica en que para entrenar una arquitectura de ese tipo desde cero, se requiere una cantidad de datos bastante considerable, debido a que muchos factores normalmente presentes en las imágenes representan ruido para lograr una detección satisfactoria, algunos de esos factores son: el color de fondo de las imágenes, la iluminación presente, la ropa que lleve el sujeto, entre otros. El método ideal para lidiar con esos factores y que el modelo pueda generalizar su aprendizaje es con un dataset de entrenamiento lo suficientemente grande, cuya obtención se saldría del alcance propuesto para este proyecto.

Durante la fase de investigación inicial para la realización de este trabajo, se encontró la existencia de MediaPipe Holistic, herramienta que representó un medio para hacer viable el desarrollo del motor con una cantidad de datos mucho más pequeña que la requerida con otras técnicas de ML; al ser un conjunto de modelos de estimación de pose para cuerpo completo, cada uno de sus componentes fue entrenado y optimizado individualmente, asegurando una resistencia bastante aceptable a los factores de ruido mencionados previamente, y además, ofrece la ventaja de trivializar los pasos de preprocesamiento necesarios después de su aplicación.

Los demás desafíos encontrados durante el desarrollo del motor fue mientras se entrenaba el modelo LSTM utilizado: la arquitectura LSTM propuesta inicialmente no resultaba capaz de aprender patrones de los datos de entrenamiento, esto quedó evidenciado por el hecho que las métricas de precisión y pérdida no variaron significativamente de sus valores iniciales, y la capacidad predictiva del modelo se mantuvo cercana a una elección aleatoria uniforme entre los 10 gestos elegidos, esto fue un claro indicador que el modelo no poseía la complejidad suficiente para adaptarse a los datos de prueba, por lo que se duplicó la cantidad de neuronas en cada capa de la red, esto resultó en una mejora respecto al escenario inicial, pero no alcanzaba un nivel de rendimiento comparable al que suele observarse en este tipo de arquitectura en trabajos similares.

Se probó duplicar nuevamente la cantidad de neuronas en todas las capas de la red, pero la mejora en este escenario fue apenas perceptible, así que dicha opción se descartó; el siguiente enfoque aplicado fue el de agregar una capa LSTM adicional luego de las 3 existentes en la red, pero esto tampoco resultó en mejoras significativas. La modificación que permitió alcanzar el resultado presentado en la sección de evaluación fue la de duplicar nuevamente el número de neuronas, pero solo en la primera capa densa luego de las capas

LSTM de la red, esto le permitió al modelo aprender los patrones presentes en los datos de entrenamiento satisfactoriamente.

Otro desafío encontrado también estuvo relacionado al entrenamiento del modelo LSTM, pero éste resultó ser de índole diferente al problema anterior, en este caso el problema encontrado era un aumento brusco del valor de la función de pérdida durante el entrenamiento, acompañado, como es de esperar, de una disminución notable de la precisión del modelo; luego de esta ocurrencia, los valores de pérdida y precisión del modelo se estancan durante todas las épocas restantes del entrenamiento, estos síntomas indican la presencia del problema conocido como gradiente explosivo (exploding gradient), que indica que durante el entrenamiento en algún punto los valores de las pesas de la red llegan a crecer descontroladamente excediendo la precisión numérica de la máquina donde se ejecuta, con el nefasto resultado de afectar negativamente el rendimiento del modelo final. Debido a la naturaleza estocástica del entrenamiento con las bibliotecas aplicadas (que mezclan los datos aleatoriamente antes de iniciar el entrenamiento) se observó que este problema no siempre ocurre, por lo que bastó con repetir el entrenamiento algunas veces hasta poder completar todas las épocas sin que ocurra el fenómeno.

Como es de esperar, aparte de la técnica mencionada para hacer frente al problema del gradiente explosivo, existen otros métodos para tratar con el mismo de manera más elegante, entre estos: agregar capas de normalización a la arquitectura de la red, que mantienen los valores numéricos en un rango donde no crecen desmesuradamente sin importar los valores de las pesas en la red; el ajuste de los parámetros del optimizador usado en el entrenamiento es otra poderosa alternativa para lidiar con este tipo de problema, el optimizador usado, Adaptive Moment Estimation (Adam), es conocido por mantener tasas de aprendizaje separadas para todas las pesas de la red, en lugar de una única tasa para todas las pesas, y ajustar dichas tasas a medida que progresa el entrenamiento, el hecho que el rendimiento del modelo se estanque luego de que ocurra el problema descrito se debe a que el optimizador disminuye la tasa de aprendizaje según aumenta el número de épocas del entrenamiento, y esto se traduce a que la optimización quede atrapada en un mínimo local no deseable de la función de pérdida, con la incapacidad de salir del mismo, justamente por la disminución en la tasa de aprendizaje que aplica el optimizador. Se puede influenciar considerablemente el comportamiento del optimizador con los parámetros del mismo.

Capítulo 5

Conclusiones y recomendaciones

El desarrollo e implementación de un motor de reconocimiento de la LSV representa un avance significativo en el ámbito de la tecnología asistiva, ofreciendo nuevas vías para facilitar la comunicación y mejorar la inclusión de la comunidad sorda. Este proyecto, a lo largo de sus distintas fases, desde el diseño inicial hasta las pruebas finales, ha abordado tanto los desafíos técnicos como los metodológicos inherentes al reconocimiento preciso y en tiempo real de las señas. Se han experimentado las capacidades y limitaciones de las tecnologías actuales de visión por computadora y aprendizaje automático, al mismo tiempo que se trató de adaptarlas al contexto específico de la LSV, dentro de los alcances planteados.

El motor de reconocimiento desarrollado no solo se presenta como una herramienta de asistencia para facilitar la interacción con las personas sordas sino también como un caso de estudio sobre la aplicación de técnicas avanzadas de procesamiento de imágenes y modelos de inteligencia artificial en la interpretación de lenguajes no verbales. La elección de la arquitectura basada en bucle, la integración de MediaPipe Holistic para el preprocesamiento de imágenes, y la comparativa entre redes neuronales convolucionales y LSTM para el reconocimiento de señas han demostrado ser fundamentales en la consecución de un sistema funcional y eficaz.

Este trabajo ha evidenciado que, a pesar de los avances en el campo de la visión por computadora y el aprendizaje profundo, la traducción automática de la lengua de señas sigue siendo una tarea compleja, que justifica el uso de las técnicas más modernas sobre redes neuronales por sobre las más básicas. En el marco de los desafíos técnicos, el más resaltante resultó ser conseguir datos de calidad para entrenar los modelos de reconocimiento empleados, aún viviendo en una época donde el volumen de información disponible públicamente es mucho mayor que el de cualquier época precedente, existen muchos casos de uso donde es un necesario un proceso correctamente orientado y ejecutado para conseguir información específica, datos en este caso, que permitan abordar un problema igual de específico, como lo es el reconocimiento de la LSV.

La experiencia acumulada durante el desarrollo de este motor de reconocimiento subraya el potencial de las soluciones tecnológicas para superar barreras de comunicación y promover la inclusión. Sin embargo, también pone de manifiesto la necesidad de continuar investigando y desarrollando, para refinar y expandir las capacidades del sistema. La colaboración entre desarrolladores de software, expertos en lengua de señas, y la comunidad sorda será crucial para asegurar que las soluciones tecnológicas no solo sean técnicamente viables sino también culturalmente apropiadas y accesibles para los usuarios finales.

5.1 Conclusiones

El desarrollo del motor de reconocimiento de la LSV ha culminado en un proyecto que no solo destaca por su innovación técnica sino también por su impacto social potencial. A través de este esfuerzo, se han obtenido varias conclusiones fundamentales que reflejan tanto los logros alcanzados como los desafíos enfrentados durante el proceso.

Primero, la efectividad del motor de reconocimiento en interpretar correctamente los gestos elegidos de la LSV subraya la viabilidad de aplicar tecnologías avanzadas de visión por computadora y aprendizaje automático en el ámbito de la comunicación accesible.

Segundo, el proyecto ha revelado la importancia crítica del diseño y selección de la arquitectura del sistema. La adopción de una arquitectura basada en bucle ha permitido un procesamiento eficiente y en tiempo real de los datos de vídeo, esencial para la traducción fluida de la lengua de señas. Esta elección arquitectónica subraya la necesidad de un enfoque sistemático y bien estructurado para el desarrollo de tecnologías de reconocimiento en tiempo real, especialmente en contextos donde la velocidad y la precisión son fundamentales.

Tercero, el proyecto ha evidenciado la complejidad inherente al reconocimiento de la lengua de señas, una lengua rica y dinámica que varía no sólo entre individuos sino también en contextos culturales. La variabilidad en la ejecución de las señas ha presentado desafíos significativos, destacando la importancia de desarrollar sistemas de reconocimiento que sean lo suficientemente flexibles para adaptarse a esta diversidad. Esto subraya la relevancia de la

participación comunitaria en el proceso de desarrollo, asegurando que el sistema sea inclusivo y representativo de la diversidad de la comunidad sorda.

Cuarto, la implementación exitosa del motor ha demostrado la utilidad de herramientas como MediaPipe Holistic en el procesamiento y análisis de gestos humanos complejos. La capacidad de esta tecnología para proporcionar datos detallados sobre la posición y el movimiento del cuerpo, las manos y la cara ha sido indispensable para el reconocimiento efectivo de señas. Esta conclusión valida la integración de tecnologías de punta en el desarrollo de soluciones para la accesibilidad lingüística.

Por último, el proyecto ha puesto en evidencia el valor de la colaboración en la investigación y desarrollo tecnológico. Particularmente gracias al joven José Gregorio Suarez, que fue tanto el creador como el referente para la mayoría de las señas elegidas en los videos recolectados, así como a su madre, que tomó el rol de intérprete en este proceso, sin ellos no habría sido posible centrar esta investigación en el contexto de la ingeniería de sistemas satisfactoriamente.

5.2 Recomendaciones a futuro

La culminación del proyecto de desarrollo del motor de reconocimiento de la LSV abre diversas avenidas para investigaciones y desarrollos futuros. Basándonos en las experiencias y aprendizajes obtenidos, se pueden establecer varias recomendaciones que buscan no solo mejorar la efectividad de tecnologías similares sino también ampliar su impacto y aplicabilidad:

Integración de Técnicas de Aprendizaje Profundo Avanzadas: Aunque el proyecto ha implementado exitosamente técnicas de visión por computadora y modelos de aprendizaje automático, la investigación en el campo del aprendizaje profundo avanza a un ritmo acelerado. Se recomienda explorar y experimentar con nuevas arquitecturas de redes neuronales, como las Redes Neuronales Convolucionales de Grafos (GCN), que pueden ofrecer una mayor precisión en el reconocimiento de patrones complejos y dinámicos inherentes a la lengua de señas.

Expansión del Conjunto de Datos de Entrenamiento: Para mejorar la generalización y precisión del motor de reconocimiento, es crucial ampliar el conjunto de datos utilizado en el entrenamiento. Esto incluye no sólo un mayor número de señas sino también una diversidad más amplia de intérpretes, abarcando variaciones en la ejecución de las señas debido a factores como edad, género y contexto cultural. La colaboración con organizaciones y comunidades sordas será vital para recopilar estos datos de manera ética y representativa.

Desarrollo de Interfaces de Usuario Inclusivas: Para asegurar la accesibilidad y usabilidad del motor de reconocimiento, se recomienda la creación de interfaces de usuario intuitivas y personalizables. Esto implica el diseño de interfaces que puedan adaptarse a las necesidades específicas de los usuarios, incluyendo ajustes en la visualización de resultados, preferencias de notificación y soporte para múltiples dispositivos.

Implementación de Retroalimentación en Tiempo Real: La incorporación de mecanismos de retroalimentación en tiempo real permitiría a los usuarios corregir o validar las interpretaciones del motor, facilitando un proceso de aprendizaje continuo para el sistema. Este enfoque de aprendizaje activo no solo mejoraría la precisión del motor a lo largo del tiempo sino también aumentaría la confianza de los usuarios en la tecnología.

Evaluación de Impacto Social y Educacional: Es fundamental llevar a cabo estudios que evalúen el impacto social y educacional del motor de reconocimiento en la comunidad sorda. Estos estudios podrían identificar áreas de mejora y nuevas oportunidades para aplicar la tecnología en contextos educativos, laborales y sociales, maximizando así su contribución a la inclusión y el bienestar de las personas sordas.

Fomento de la Colaboración Interdisciplinaria: Finalmente, se alienta la continuidad y expansión de la colaboración entre ingenieros, lingüistas, psicólogos, educadores y miembros de la comunidad sorda. Esta colaboración interdisciplinaria es esencial para el desarrollo de soluciones tecnológicas que sean técnicamente sólidas, culturalmente sensibles y socialmente impactantes.

Bibliografía

[1] Stephen W. Littlejohn, Karen A. Foss, John G. (2016). Theories of Human Communication.

[2] Garcia, & Alarcon. (2016). Real-time American Sign Language Recognition with Convolutional Neural. Stanford University.

[3] Ibrahim, Nada & Zayed, Hala & Selim, Mazen. (2019). Advances, Challenges, and Opportunities in Continuous Sign Language Recognition. Journal of Engineering and Applied Sciences. 15. 1205-1227. 10.36478/jeasci.2020.1205.1227.

[4] Dominguez, M (1996). Los verbos de la LSV, fundamentos para comprender la morfología verbal de una lengua de señas. [Tesis de maestría]. Universidad de los Andes, Biblioteca Tulio Febres Cordero, Universidad de Los Andes, Mérida, Venezuela, digitalizado por SERVIULA.

[5] Lourdes Pietrosemoli. (1989) Primer seminario de lingüística de la lengua de señas venezolana (l.s.v). In I SEMINARIO DE LINGÜÍSTICA DE LA LENGUA DE SEÑAS VENEZOLANA (LSV), Mérida, Venezuela.

[6] T. Wang, G. Song, W. Ni and Q. Zeng. (2023). "MIFD-Net : a hand gesture recognition model based on feature fusion of MLP and CNN".

[7] Wu X Y. (2020). A hand gesture recognition algorithm based on DC-CNN. Multimedia Tools and Applications. 79(13): 9193-205.

[8] Wadhawan A, and Kumar P. (2020). Deep learning-based sign language recognition system for static signs. Neural computing and applications. 32(12): 7957-68.

[9] Saqib S, Ditta A, et al. (2020). Intelligent dynamic gesture recognition using CNN empowered by edit distance. Comput. Mater. Contin. 66: 2061–76.

[10] Neethu P S, Suguna R and Sathish D. (2020). An efficient method for human hand gesture detection and recognition using deep learning convolutional neural networks. Soft Computing. 24(20): 15239-248.

[11] Blanco, M. (2016). Desarrollo de un motor de traducción del lenguaje hablado a lenguaje de señas venezolano. (Caso: Ingeniería de Sistemas). [Tesis de grado]. Universidad de los Andes.

[12] LeCun, Y., Bengio, Y., Bottou, L. & Haffner, P. (1998). Gradient-Based Learning Applied to Document Recognition. Proceedings of the IEEE, 86(11), 2278-2324.

[13] A.Oviedo, H. Rumbos y Y. Pérez H. (2004) „El estudio de la Lengua de Señas Venezolana“: En: F. Freites y F.J. Pérez (eds.) Las disciplinas lingüísticas en Venezuela. Maracaibo: Universidad Católica Cecilio Acosta, pp. 201233.

[14] Pérez, Y., (2007). La norma en la lengua de señas venezolana. Sapiens. Revista Universitaria de Investigación, 8(2), 105-121.

[15] Federación Venezolana de Sordos FEVENSOR. (1999). Manual de Lengua de Señas Venezolana.

[16] Wang, Robert & Popovic, Jovan. (2009). Real-time hand-tracking with a color glove. ACM Trans. Graph.. 28. 10.1145/1576246.1531369.

[17] RAGHUVEERA, DEEPTHI, MANGALASHRI and AKSHAYA. (2019). A depth-based Indian Sign Language recognition using Microsoft Kinect. Department of Computer Science and Engineering, College of Engineering Guindy Campus, Anna University, Chennai 600025, India.

[18] Arai, Kohei & Kapoor, Supriya. (2020). Advances in Computer Vision Proceedings of the 2019 Computer Vision Conference (CVC), Volume 2: Proceedings of the 2019 Computer Vision Conference (CVC), Volume 2. 10.1007/978-3-030-17798-0.

[19] Mahony, N.O., Campbell, S., Carvalho, A., Harapanahalli, S., Velasco-Hernández, G.A., Krpalkova, L., Riordan, D., & Walsh, J. (2019). Deep Learning vs. Traditional Computer Vision. Computer Vision Conference.

[20] Murhij, Y., Serebrenny, V. (2021). Hand Gestures Recognition Model for Augmented Reality Robotic Applications. In: Ronzhin, A., Shishlakov, V. (eds) Proceedings of 15th International Conference on Electromechanics and Robotics "Zavalishin's Readings". Smart Innovation, Systems and Technologies, vol 187. Springer, Singapore. https://doi.org/10.1007/978-981-15-5580-0_15

[21] González, R.C., & Woods, R.E. (2018). Digital Image Processing. Pearson.

[22] Al Bovik. (2009). The Essential Guide to Video Processing. ELSEVIER.

[23] M. Al-Qurishi, T. Khalid and R. Souissi, "Deep Learning for Sign Language Recognition: Current Techniques, Benchmarks, and Open Issues," in IEEE Access, vol. 9, pp. 126917-126951, 2021, doi: 10.1109/ACCESS.2021.3110912.

[24] Z. Li, F. Liu, W. Yang, S. Peng and J. Zhou, "A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects," in IEEE Transactions on Neural Networks and Learning Systems, vol. 33, no. 12, pp. 6999-7019, Dec. 2022, doi: 10.1109/TNNLS.2021.3084827.

[25] Wang J, Ma Y, Zhang L, Gao RX (2018) Deep learning for smart manufacturing: Methods and applications. J Manuf Syst 48:144–156. https://doi.org/10.1016/J.JMSY.2018.01.003

[26] Balaji, S. (2020). Estructura de una CNN. [Imagen en la web]. Binary Image classifier CNN using TensorFlow. Medium. Disponible en Binary Image classifier CNN using TensorFlow | by Sai Balaji | Techiepedia | Medium

[27] Creswell, J. W. (2009). Research design: Qualitative, quantitative, and mixed methods approaches. Sage publications.

[28] Manning C., Raghavan P. & Schütze H.(2009). An Introduction to Information Retrieval. Cambridge University Press Cambridge, England.

[29] Smith, M. (2015). Research methods in accounting. Sage Publications.

[30] Carmines, E., Zeller, R. (1979). Reliability and validity assessment. Sage Publications.

[31] Saravanan, C. (2010). Color Image to Grayscale Image Conversion. Second International Conference on Computer Engineering and Applications.

[32] Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. Neural Computation, 9(8), 1735-1780.

[33] Gers, F. A., Schmidhuber, J., & Cummins, F. (2000). Learning to Forget: Continual Prediction with LSTM. Neural Computation, 12(10), 2451-2471.

[34] Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press.

[35] Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y. (2014). Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation.

[36] Grishchenko I., Bazarevsky V. (2020). MediaPipe Holistic — Simultaneous Face, Hand and Pose Prediction, on Device. Disponible en MediaPipe Holistic — Simultaneous Face, Hand and Pose Prediction, on Device – Google Research Blog.

[37] Singh, A. K., Kumbhare, V. A., & Arthi, K. (2022). Real-Time Human Pose Detection and Recognition Using MediaPipe.

[38] Velmathi G, Kaushal G (2023). Indian Sign Language Recognition Using Mediapipe Holistic. ArXiv.

[39] DroidScript. JavaScript IDE. Disponible en DroidScript – JavaScript IDE

[40] Martins, L. ¿Qué es la metodología Kanban y cómo funciona? (2022) • Asana. Disponible en ¿Qué es la metodología Kanban y cómo funciona? [2022] • Asana

[41] Vasilev, I., Slater, D., & Spadon, G. (2019). Python Deep Learning: Exploring deep learning techniques and neural network architectures with PyTorch, Keras, and TensorFlow (2nd ed.). Packt Publishing.

[42] Bass, L., Clements, P., & Kazman, R. (2012). Software Architecture in Practice (3rd ed.). Addison-Wesley Professional.

[43] Shaw, M., & Garlan, D. (1996). Software Architecture: Perspectives on an Emerging Discipline. Prentice-Hall, Inc.

[44] Bradski, G. (2000). The OpenCV Library. Dr. Dobb's Journal of Software Tools.

[45] TensorBoard | TensorFlow. Disponible en TensorBoard | TensorFlow