



**FACULTAD DE INGENIERÍA, ARQUITECTURA Y
URBANISMO**

ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS

TESIS

**Implementación de un modelo de aprendizaje
profundo para la traducción de lenguajes de señas**

**PARA OPTAR EL TÍTULO PROFESIONAL DE
INGENIERO(A) DE SISTEMAS**

Autor(es)

**Bach. Santisteban Ostos Andy Josue
ORCID <https://orcid.org/0000-0001-5972-5995>
Bach. Pacheco Contreras Nicolette Isis
ORCID <https://orcid.org/0000-0002-7837-9132>**

Asesor(a):

**DR. Forero Vargas Manuel Guillermo
ORCID <https://orcid.org/0000-0001-9972-8621>**

**Línea de Investigación
Infraestructura, Tecnología y Medio Ambiente**

Pimentel – Perú

2024

**Implementación de un modelo de aprendizaje profundo para la traducción
de lenguajes de señas**

Aprobación del jurado

MG. ASENJO CARRANZA ENRIQUE DAVID

Presidente del Jurado de Tesis

MG. QUINTEROS GONZALES HERMES MARINO

Secretario del Jurado de Tesis

MG. ALVA ZAPATA JULIANA DEL PILAR

Vocal del Jurado de Tesis

DECLARACIÓN JURADA DE ORIGINALIDAD

Quienes suscriben la DECLARACIÓN JURADA, somos **Egresados** del Programa de Estudios de **Ingeniería de Sistemas** de la Universidad Señor de Sipán S.A.C, declaramos bajo juramento que somos autores del trabajo titulado:

IMPLEMENTACIÓN DE UN MODELO DE APRENDIZAJE PROFUNDO PARA LA TRADUCCIÓN DE LENGUAJES DE SEÑAS

El texto de mi trabajo de investigación responde y respeta lo indicado en el Código de Ética del Comité Institucional de Ética en Investigación de la Universidad Señor de Sipán, conforme a los principios y lineamientos detallados en dicho documento, en relación con las citas y referencias bibliográficas, respetando el derecho de propiedad intelectual, por lo cual informo que la investigación cumple con ser inédito, original y autentico.

En virtud de lo antes mencionado, firman:

Santisteban Ostos Andy Josue	DNI: 71985693	
Pacheco Contreras Nicolette Isis	DNI: 77416740	

Pimentel, 02 de Julio de 2023.

NOMBRE DEL TRABAJO

Implementación de un modelo de aprendizaje profundo para la traducción de lenguajes de señas

AUTOR

Andy Josue / Nicolette Isis Santisteban Ostos / Pacheco Contreras

RECUENTO DE PALABRAS

12284 Words

RECUENTO DE CARACTERES

67474 Characters

RECUENTO DE PÁGINAS

60 Pages

TAMAÑO DEL ARCHIVO

5.2MB

FECHA DE ENTREGA

Apr 29, 2024 3:42 PM GMT-5

FECHA DEL INFORME

Apr 29, 2024 3:43 PM GMT-5**● 14% de similitud general**

El total combinado de todas las coincidencias, incluidas las fuentes superpuestas, para cada base de datos.

- 8% Base de datos de Internet
- Base de datos de Crossref
- 10% Base de datos de trabajos entregados
- 2% Base de datos de publicaciones
- Base de datos de contenido publicado de Crossref

● Excluir del Reporte de Similitud

- Material bibliográfico
- Coincidencia baja (menos de 8 palabras)
- Material citado

Dedicatoria

Dedico esta tesis a Dios, quien me da amor, sabiduría, inteligencia, salud y paciencia. También a mis padres por el apoyo incondicional económico, moral y valores que pusieron en mi para lograr esta meta.

Nicolette Isis Pacheco Contreras

La presente tesis se la dedico en primer lugar a Dios, seguido de mis padres, hermanos, novia y amigos que siempre estuvieron a mi lado apoyándome a salir adelante de manera incondicional.

Andy Josue Santisteban Ostos

Agradecimientos

Agradezco infinitamente a Dios, a mi familia que de alguna forma contribuyeron a la realización de esta tesis durante mi trayectoria académica y personal.

También agradecer a cada docente de la Facultad de Ingeniería, Arquitectura y Urbanismo de la Universidad Señor de Sipán por brindarme los recursos necesarios, conocimientos enriquecedores y sugerencias valiosas para mejorar significativamente el logro de esta tesis.

Nicolette Isis Pacheco Contreras

Primeramente, agradezco a Dios por todo lo que me ha permitido vivir
A mis Padres Hugo Santisteban Radaeli y Victoria Milagros Ostos Camacho por apoyarme en los momentos difíciles y guiarme por toda mi vida, a mis hermanos Hugo, Maria, David y Daniel por siempre ser esos hermanos y compañeros de vida.

A mis profesores de toda la carrera que me enseñaron en cada clase los conocimientos que hicieron posible esta tesis.

Andy Josue Santisteban Ostos

Índice

Dedicatoria	IVI
Agradecimientos	VI
Índice de Figuras y Tablas	VI
Resumen	VI
Abstract	XII
I. INTRODUCCIÓN	13
1.1. Realidad problemática	13
1.2. Formulación del problema.....	17
1.3. Hipótesis.....	17
1.4. Objetivos.....	17
1.5. Teorías relacionadas al tema	18
II. MATERIAL Y MÉTODO.....	33
2.1 Tipo y Diseño de Investigación	33
2.2 Variables, Operacionalización.....	33
2.3 Población de estudio, muestra, muestreo y criterios de selección.....	36
2.4 Técnicas e instrumentos de recolección de datos, validez y confiabilidad.....	36
2.5 Procedimiento de análisis de datos.....	36
2.6 Criterios éticos	39
III. RESULTADOS Y DISCUSIÓN.....	40
3.1. Resultados.....	40
3.2. Discusión	50
3.3. Aporte de la investigación.....	51

IV.	CONCLUSIONES Y RECOMENDACIONES	70
4.1.	Conclusiones	70
4.2.	Recomendaciones	72
	REFERENCIAS	74
	ANEXOS	79

Índice de Figuras y Tablas

Fig. 1. Descripción gráfica de la propagación hacia atrás	22
Fig. 2. Submuestreo cromático de 2x2 reduciendo a la mitad	24
Fig. 3. Labellmg para la tarea de anotación	26
Fig. 4. Diagrama de procesamiento de Tensorflow	27
Fig. 5. Uso de Tensorflow con WebGL.....	28
Fig. 6. Arquitectura You Only Look Once (YOLO).....	29
Fig. 7. Interacción WEBSOCKET	30
Fig. 8. Arquitectura de Node.js del uso de hilos.	32
Fig. 9. Diagrama de uso de React.js y conexión con servicios.	33
Fig. 10. Resultados de mAP de los modelos implementados	41
Fig. 11. (a) Representa la matriz de precisión del modelo YOLO v5 (b) Representa la precisión del modelo YOLO v5 personalizado (c) Gráfico de barras que representa visualmente la precisión del modelo MobileNet v2.....	42
Fig. 12. (a) Representa la matriz de recuperación del modelo YOLO v5 (b) Representa la recuperación del modelo YOLO v5 personalizado (c) Gráfico de barras que representa visualmente la recuperación del modelo MobileNet v2.....	43
Fig. 13. Resultados de F1-Confidence Curva de los modelos implementados.	44
Fig. 14. Tiempo de ejecución del modelo YOLO v5 personalizado para el dataset de validación.....	45
Fig. 15. Tiempo de ejecución del modelo de YOLO v5 para el dataset de validación.....	46
Fig. 16. Resultados del tiempo de procesamiento.....	47
Fig. 17. Resultados del tiempo de procesamiento estadístico.....	48
Fig. 18. Resultados del modelo basado en YOLOv5.....	49
Fig. 19. Imágenes con formato .jpg adquiridas por el algoritmo.	52
Fig. 20. Formato txt adquirido por el Labellmg.....	53
Fig. 21. MODELO YOLOV5	59

Fig. 22. Implementación del modelo MobileNet V2.	60
Fig. 23. Configuraciones para WebGL en el navegador.	64
Fig. 24. Modelo de YOLO Exportado para Tensorflow.js.....	65
Fig. 25. Archivos binarios exportados con el modelo.....	65
Fig. 26. Ubicación de aceleración por hardware.....	68
Fig. 27. Carga de modelo en la aplicación web con diferentes usuarios.....	68
Fig. 28. Carga de modelo en la aplicación web con diferentes usuarios.....	69
Fig. 29. Aplicación web "" - transmisión de videos en diferentes aplicaciones.	69
Fig. 30. Aplicación web "https://signape.onrender.com/" - Formando palabras con letras. .	69
Fig. 31. Aplicación web "https://signape.onrender.com/" - Formando palabras con letras ..	70
Fig. 32. Procesamiento de número 0 con Labellmg	94
Fig. 33. Resultado de número 0 en XML	95
Fig. 34. Procesamiento de la palabra "piel" con Labellmg	95
Fig. 35. Muestra de imágenes procesadas por Labellmg	96
Fig. 36. Resultado de data en formato numpy (.npy).....	97
Fig. 37. Gráfico de Matriz de confusión Modelo YOLO V5	102
Fig. 38. Gráfico de Matriz de confusión Modelo YOLO V5 personalizado	102
Fig. 39. Gráfico de Matriz de confusión Modelo MobileNet V2	103

TABLA I. VARIABLES, OPERACIONALIZACIÓN	34
TABLA II. RESULTADOS DE VALOR MEDIO DE PRECISIÓN MEDIA.....	40
TABLA III. RESULTADOS DE PRECISIÓN.	41
TABLA IV. RESULTADOS DE RECUPERACIÓN.....	42
TABLA V. RESULTADOS DE CÁLCULO DE LA SENSIBILIDAD F1-CONFIDENCE.....	44
TABLA VI. FORMATOS DE ARCHIVOS DE LOS MODELOS	53
TABLA VII. DISTRIBUCIÓN DE LAS CARPETAS PARA EL ENTRENAMIENTO.....	54
TABLA VIII. MODELOS QUE PRESENTAN MEJORES RESULTADOS.	55
TABLA IX. HIPER PARÁMETROS DE YOLOV5.....	57
TABLA X. HIPER PARÁMETROS DE MOBILE NET V2.....	60
TABLA XI. MODELOS DE RED NEURONAL DE APRENDIZAJE PROFUNDO.	84
TABLA XII. LISTA DE LENGUAJE DE SEÑAS	90

Resumen

En el Perú, se ha registrado un elevado número de personas con discapacidad auditiva y dificultad para oír, de acuerdo con el Censo Nacional de Población (INEI-2017), con más de 232,000 individuos en ambas categorías. Lamentablemente, a muchas de estas personas se les presentan barreras para acceder a servicios públicos y educativos debido a la escasez de docentes e intérpretes capacitados en Lengua de Señas Peruana. Estas barreras dificultan su acceso a una educación inclusiva y limitan su participación en la sociedad. Por otro lado, se encuentran desafíos computacionales en el entrenamiento y la ejecución de modelos de aprendizaje profundo debido a su complejidad y a la necesidad de recursos significativos. En función de los datos previamente expuestos, el objetivo de esta investigación consistió en implementar un sistema web basado en modelos de aprendizaje profundo para la traducción del lenguaje de señas peruana, con el propósito de lograr la detección en tiempo real de letras, a fin de brindar una solución práctica para mejorar la comunicación entre quienes padecen de discapacidad auditiva y los demás miembros de la sociedad. Para lograr dicho objetivo, se decidió utilizar el modelo YOLO en el sistema de reconocimiento de lenguaje de señas debido a los resultados que ofrece. El modelo YOLO alcanzó una precisión del 97.6%, valor medio de precisión media 90.2% y una puntuación F1-Confidence del 89.18%, lo que destaca su capacidad para detectar de manera precisa los gestos del lenguaje de señas peruano. La implementación del sistema basado en el modelo YOLO proporciona una solución innovadora y efectiva para la detección en tiempo real del Lenguaje de Señas Peruano (LSP) en entornos web. Esta implementación brinda oportunidades para mejorar la comunicación e inclusión de las personas que utilizan este lenguaje.

Palabras Clave: Aprendizaje profundo, lenguaje de señas peruana, redes neuronales, reconocimiento de gestos, accesibilidad para personas sordas.

Abstract

In Peru, there is a high number of people with hearing impairment and difficulty hearing, according to the National Population Census (INEI-2017), with more than 232,000 individuals in both categories. Unfortunately, many of these people are presented with barriers to access public and educational services due to the shortage of teachers and interpreters trained in Peruvian Sign Language. These barriers hinder their access to inclusive education and limit their participation in society. On the other hand, there are computational challenges in the training and execution of deep learning models due to their complexity and the need for significant resources. Based on the data previously presented, the objective of this research was to implement a web system based on deep learning models for the translation of Peruvian sign language, with the purpose of achieving real-time detection of letters, in order to provide a practical solution to improve communication between those who suffer from hearing impairment and other members of society. To achieve this goal, it was decided to use the YOLO model in the sign language recognition system because of the results it provides. The YOLO model achieved an accuracy of 97.6%, mean average accuracy value 90.2% and F1-Confidence score of 89.18%, which highlights its ability to accurately detect Peruvian sign language gestures. The implementation of the system based on the YOLO model provides an innovative and effective solution for real-time detection of Peruvian Sign Language (PSL) in web environments. This implementation provides opportunities to improve communication and inclusion of people who use this language.

Keywords: Deep learning, Peruvian sign language, neural networks, gesture recognition, accessibility for deaf people

I. INTRODUCCIÓN

1.1. Realidad problemática

De acuerdo con la Organización Mundial de la Salud (OMS), se estima que hay 360 millones de personas en todo el mundo que experimentan pérdida de audición. Esto incluye a aquellas personas que tienen discapacidad o deficiencia auditiva [1], Además, según la estimación de la Federación Mundial de Sordos, se estima que alrededor de 70 millones de personas en todo el mundo utilizan el lenguaje de señas como su lengua materna o primera lengua [2].

Una persona sorda (PS), es aquella que presenta una deficiencia en la audición, lo que implica que no puede escuchar o solo puede hacerlo de manera limitada. Además, puede tener dificultades para comprender el habla a través del oído. Por otro lado, una persona que tiene o adquiere una discapacidad auditiva (PDA) presenta una deficiencia en diferentes niveles, como leve, moderada, grave o profunda en la audición, la cual podría o no corregirse con amplificación [3].

En la actualidad, la educación de las PDA y PS representa un desafío a nivel global. Esto se debe a la falta de profesores que estén familiarizados con la cultura sorda y que tengan experiencia en el lenguaje de señas o de signos. Como resultado, se genera un aislamiento que impide que estas personas participen plenamente y sean capaces de aprender al mismo ritmo que sus compañeros [4]. Según la Federación Mundial de Sordos (FMS), se estima que hay aproximadamente 56 millones de personas sordomudas en todo el mundo que no tienen acceso a ningún tipo de educación [5].

Las PDA y PS pueden enfrentar diversos desafíos en el ámbito educativo debido a la pérdida de audición, lo cual puede afectar su pensamiento, habla, lenguaje, conducta, desarrollo social y emocional. Estos factores pueden provocar una disminución en el logro de sus aprendizajes [6]. Además, la dificultad para comunicarse con los demás puede generar problemas adicionales, como ansiedad, aislamiento social y depresión [7]. Para abordar los

desafíos de comunicación que enfrentan las PS, se utilizan las lenguas de señas. Estas lenguas pueden ser aprendidas y utilizadas tanto por las PDA como por sus familiares, amigos e intérpretes de lengua de señas [8].

La pandemia de COVID-19 ha tenido efectos en las PD y PDA en diversos aspectos. Un estudio publicado en el National Center for Biotechnology Information (NCBI) señala que la pandemia ha impactado la vida cotidiana de estas personas, afectando la comunicación, el acceso a la información, la atención médica y el cambio del modelo de enseñanza presencial a la modalidad de enseñanza a distancia [9]. Sin embargo, la falta de acceso a la mayoría de las herramientas en línea, junto con la escasez de intérpretes de lenguaje de señas, ha creado barreras significativas para los estudiantes sordos durante las clases virtuales [10]

Según el Censo Nacional de Población (INEI-2017), en Perú existen 232,176 personas con dificultad para oír y más de 232,000 personas con discapacidad auditiva. Lamentablemente, muchas de estas personas no pueden acceder a servicios públicos y educativos debido a la falta de docentes e intérpretes de Lengua de Señas Peruana (LSP), situación que se ha agravado durante la pandemia del COVID-19 [11]. En Perú, las personas más afectadas por la pandemia fueron las personas sordas y las PDA, debido al uso generalizado de mascarillas faciales. Esta medida de protección dificulta la lectura de labios, que es una herramienta clave en la comunicación para estas personas, y también dificulta la comprensión del lenguaje de señas [12]. Además, la pandemia ha tenido un impacto significativo en la educación de los niños sordos, lo que ha llevado a retrasos no solo en los estudios en todo el país, sino también en la interacción con otros niños sordos para aprender el lenguaje de señas [13].

En la actualidad, se han desarrollado tecnologías relacionadas con el lenguaje de señas con el objetivo de facilitar la comunicación entre personas con discapacidad auditiva, personas sordas y oyentes. Por ejemplo, Signslator es una aplicación web que traduce del español al lenguaje de señas utilizando intérpretes virtuales. Esta herramienta permite difundir, enseñar y fortalecer el aprendizaje del lenguaje de señas en personas con discapacidad auditiva y dificultades en el habla. Además, también funciona como un

diccionario, brindando definiciones y ejemplos de señas para facilitar la comunicación y el acceso a información para la comunidad sorda [14]. Otro traductor de lenguaje de señas es el Showleap, que permite convertir los gestos de la lengua de señas en texto y voz. Este proyecto obtuvo el primer puesto en la categoría Accesibilidad TIC de los Premios Vodafone a la Innovación 2018. Showleap utilizó técnicas de aprendizaje profundo y un programa compuesto por tres redes neuronales para procesar los gestos y proporcionar una traducción significativa [15] .

Otro software de reconocimiento de lenguaje de señas utiliza el dispositivo Leap Motion para detectar los movimientos realizados por las manos del usuario y reproducirlos en un ordenador. El dispositivo Leap Motion recibe las señales de infrarrojos y las procesa para transformarlas en letras del lenguaje de señas. De esta manera, se logra una traducción visual y comprensible de los gestos en tiempo real [16].

Los modelos de aprendizaje profundo son comúnmente utilizados para el reconocimiento del lenguaje de señas. Esto les permite identificar imágenes y gestos de la mano, aprendiendo características espaciales de las imágenes capturadas por cámaras, así como de conjuntos de datos que incluyen imágenes, datos de gestos de la mano y las etiquetas correspondientes a los signos del lenguaje de señas [17]

Los modelos más destacados y utilizados en el campo del aprendizaje profundo son las redes neuronales convolucionales (CNN), debido a su capacidad ampliamente reconocida para el procesamiento de imágenes y la extracción de características. Estas redes son especialmente eficientes en la tarea de reconocimiento de patrones [17] [18] [19] [20]. El modelo CNN es eficaz para extraer características espaciales de las imágenes estáticas. Al combinar 2DCNN, 3DCNN y LSTM, el modelo fue capaz de predecir los gestos de lenguaje de señas [21].

Se han empleado redes neuronales convolucionales (CNN) y redes neuronales recurrentes (RNN) [22] , en conjunto con diversos modelos como ResNet18, MobileNet_V2, EfficientNet_B1, AlexNet, ResNet50, EfficientNet-B7 e InceptionV3 [23], así como VGGNet-16 y VGGNet-19 [24] , junto al modelo YOLO [25], para llevar a cabo la extracción de

características de los datos de imagen.

Otro aspecto relevante es la importancia de los modelos CNN, como MobileNetV2 y ResNet18, debido a su destacado rendimiento y eficiencia en la extracción de características espaciales. Esto los posiciona como opciones apropiadas para aplicaciones de reconocimiento de gestos en videos de lenguaje de señas [26].

Se destaca que YOLO ha sido aplicado en la detección de peatones, localización de placas de matrícula de vehículos, aplicaciones médicas y reconocimiento de lenguaje de señas, entre otros usos [25]. YOLO es reconocido por su capacidad para procesar video en tiempo real a una velocidad de 28.9 cuadros por segundo [27].

No obstante, el reconocimiento de los lenguajes de señas presenta desafíos debido al análisis de conjuntos finitos de formas de mano que varían rápidamente en cada segundo, lo que dificulta su reconocimiento preciso [28]. Además, el ruido de fondo en la imagen [29] o la presencia de fondos complejos con ruido semántico, junto con variaciones en la iluminación, pueden tener un impacto significativo en el procesamiento de imágenes [30].

Se evaluaron los modelos de aprendizaje profundo para determinar su precisión en el reconocimiento de los lenguajes de señas, considerando los resultados del entrenamiento experimental. En resumen, el rendimiento sobresaliente en el reconocimiento de lenguajes de señas ha sido demostrado por el modelo ResNet 18, el cual exhibe resultados destacados en tareas de clasificación [31]. Por otro lado, el modelo de red YOLO también ha demostrado su capacidad para traducir de manera eficiente 24 signos del conjunto de datos ASL (American Sign Language) en tiempo real, además de dos signos adicionales. Estos logros evidencian la eficacia y rapidez del modelo en el reconocimiento y traducción de los signos del lenguaje de señas [25].

Para alcanzar los objetivos del presente trabajo de investigación, se llevó a cabo una evaluación del tiempo que el proceso tarda en completarse, el consumo promedio del CPU y la Escala de Satisfacción en el sistema web. Además, se explica que el sistema emplea modelos de aprendizaje profundo para desarrollar una solución de traducción al lenguaje de señas. Esta solución tiene como objetivo principal mejorar la comprensión y la expresión de

quienes utilizan el lenguaje de señas, tanto personas con discapacidad auditiva como sordas y oyentes, a través de la comunicación por medio de las letras del abecedario de lenguaje de señas. El propósito es superar las barreras de comunicación existentes y proporcionar una forma más efectiva de interactuar y expresarse en su entorno cotidiano para este grupo de personas.

1.2. Formulación del problema

¿De qué manera se puede implementar un modelo de aprendizaje profundo para la traducción de lenguaje de señas peruano en un sistema web?

1.3. Hipótesis

La implementación de una CNN entrenada con datos del lenguaje de señas peruano permitirá la traducción en tiempo real de las señales capturadas por la cámara del usuario en un sistema web, mejorando la accesibilidad y la comunicación para personas con discapacidades auditivas en Perú.

1.4. Objetivos

La implementación de una CNN entrenada con datos del lenguaje de señas peruano permitirá la traducción en tiempo real de las señales capturadas por la cámara del usuario en un sistema web, mejorando la accesibilidad y la comunicación para personas con discapacidades auditivas en Perú.

Objetivo general

Implementar un sistema web de video llamadas basado en aprendizaje profundo para la traducción de letras del lenguaje de señas peruano.

Objetivos específicos

- Adquirir una base de datos que contenga imágenes de letras de lenguaje de señas.
- Establecer e implementar modelos de aprendizaje profundo para la traducción del lenguaje de señas.
- Desarrollar un sistema web de videollamadas que permita la traducción de lenguaje de señas a lenguaje escrito empleando el modelo profundo desarrollado.

- Analizar los resultados de las técnicas implementadas de lenguaje de señas.

1.5. Teorías relacionadas al tema

1.5.1. Lenguaje de señas

El lenguaje de señas (LS), es un derecho lingüístico fundamental para todas las personas que tienen dificultades o discapacidades auditivas. Este derecho surge de la necesidad de establecer una comunicación efectiva entre los interlocutores, permitiendo a las personas sordas o con problemas de audición expresarse, comprender y participar plenamente en la sociedad.

Actualmente se han identificado y registrado alrededor de 121 variedades de lenguajes de señas utilizados por comunidades sordas en todo el mundo Anexo 6. Siendo las más conocidas el lenguaje de señas americano (ASL), británico (BSL), francés (LSF), japonés (JSL), alemán (DGS), italiano (LIS), chino (CSL) y el lenguaje de señas ruso (RSL), entre muchos otros.

Entre estas variedades se encuentra el lenguaje de señas peruano, que es utilizado por la comunidad sorda en Perú. Es importante destacar que estas variedades de lenguajes de señas son utilizadas por personas sordas para comunicarse y expresarse en su vida diaria. Se estima que el número total de personas sordas que utilizan estos lenguajes de señas supera los 700 millones a nivel mundial. Estas cifras demuestran la diversidad y la importancia de los lenguajes de señas en la comunicación y la cultura de las personas sordas en diferentes países y regiones [32]. La diferencia principal radica en que los lenguajes de señas caseros son desarrollados y utilizados en entornos familiares o en comunidades pequeñas con pocos miembros que necesitan comunicarse entre sí. Estos lenguajes de señas pueden ser creados de manera espontánea y adaptarse a las necesidades y preferencias de la familia o grupo en particular [33].

1.5.2. Inteligencia Artificial

La inteligencia artificial abarca una amplia variedad de temas y aplicaciones, incluyendo el desarrollo de sistemas inteligentes para automatizar tareas complejas, la creación de agentes capaces de interactuar y tomar decisiones. Además, abarca investigaciones en campos como el procesamiento de lenguaje natural, la visión computacional y la robótica. También se exploran algoritmos de optimización para resolver problemas eficientemente, se aplican técnicas de aprendizaje profundo para el análisis de datos y toma de decisiones, y se utilizan en campos como medicina, industria, seguridad y entretenimiento [34] .

1.5.3. Aprendizaje automático (Machine learning)

El Aprendizaje automático es un campo de la inteligencia artificial que se enfoca en el desarrollo de algoritmos que capacitan a un sistema para aprender de los datos proporcionados, sin requerir una programación explícita para cada tarea específica [32]. Según Alpaydin, el propósito fundamental del aprendizaje automático es “desarrollar algoritmos que permitan a las máquinas mejorar su rendimiento en una tarea específica a través de la experiencia” (p. 3). Por su parte,[33] definen el aprendizaje automático como “un conjunto de métodos que se utilizan para construir modelos a partir de datos” (p. 1). Estos modelos pueden ser utilizados para hacer predicciones o tomar decisiones en base a nuevos datos de entrada.

En el aprendizaje automático, los datos de entrada son utilizados para entrenar un modelo matemático que pueda hacer predicciones o clasificaciones sobre nuevos datos. Uno de los algoritmos más utilizados en el aprendizaje supervisado es la regresión lineal, que busca encontrar la mejor recta que se ajuste a los datos de entrada [35]. Otros algoritmos populares incluyen los árboles de decisión y las redes neuronales [33].

1.5.4. Redes neuronales

Estas son un modelo matemático basado en la estructura y función de las neuronas biológicas. En el contexto del Aprendizaje profundo (Deep Learning), las redes neuronales son el modelo básico de aprendizaje automático. Las fórmulas más comunes utilizadas en redes neuronales están relacionadas en calcular el error y actualizar los pesos de conexión [36]

1.5.4.1. Función de pérdida

Es una medida de qué tan bien la red neuronal está haciendo predicciones en los datos de entrenamiento. Una de las fórmulas más utilizadas es el error cuadrático medio o MSE, por sus siglas en inglés:

$$MSE = \frac{1}{n} * \sum_{i=1}^n (y_{pred_i} - y_{true_i})^2$$

Donde n es el número que se tiene de ejemplos para el entrenamiento, y_i es el valor verdadero de la salida y \hat{y}_i es el valor que la red neuronal predijo.

1.5.5. Aprendizaje profundo

El aprendizaje profundo utiliza redes neuronales para el aprendizaje automático con el fin de aprender características jerárquicas de los datos de entrada. El término "profundo" se refiere a la profundidad de la red, es decir, al número de capas ocultas que tiene. La teoría detrás del aprendizaje profundo se basa en el teorema de aproximación universal, que establece que una red neuronal profunda puede aproximar cualquier función continua en un espacio de dimensión finita [36]

La fórmula más utilizada en el aprendizaje profundo es la de la propagación hacia atrás (backpropagation):

1.5.5.1. Propagación hacia atrás (Backpropagation)

Esta fórmula se utiliza para calcular el gradiente de la función de pérdida con respecto a los pesos de las conexiones en una red neuronal profunda. La propagación hacia atrás consiste en propagar el error desde la salida de la red hacia las capas ocultas, utilizando la regla de la cadena para calcular el gradiente:

$$\frac{\partial L}{\partial W} = \left(\frac{\partial L}{\partial o}\right) * \left(\frac{\partial o}{\partial z}\right) * \left(\frac{\partial z}{\partial w}\right)$$

Donde L es la función de pérdida, o es la salida de la red neuronal y z es la entrada de la función de activación de la neurona i.

1.5.5.2. Regla de Cadena

La regla de cadena se utiliza para calcular el gradiente que contiene la función de pérdida con respecto a los pesos de las conexiones:

$$\frac{\partial L}{\partial W} = \left(\frac{\partial L}{\partial o}\right) * \left(\frac{\partial o}{\partial w}\right)$$

Donde L hace referencia a la función de pérdida, o es la salida de la red neuronal y w son los pesos de las conexiones.

1.5.6. Redes neuronales Recurrentes (RNN):

Una red neuronal recurrente (RNN) es un tipo de estructura de red neuronal diseñada para trabajar con secuencias de datos. En contraste con las redes neuronales Feedforward estándar, las RNN incluyen conexiones de retroalimentación, lo que posibilita que la salida de una capa se convierta en la entrada de la siguiente. Esta capacidad les permite capturar y modelar las dependencias temporales en los datos.

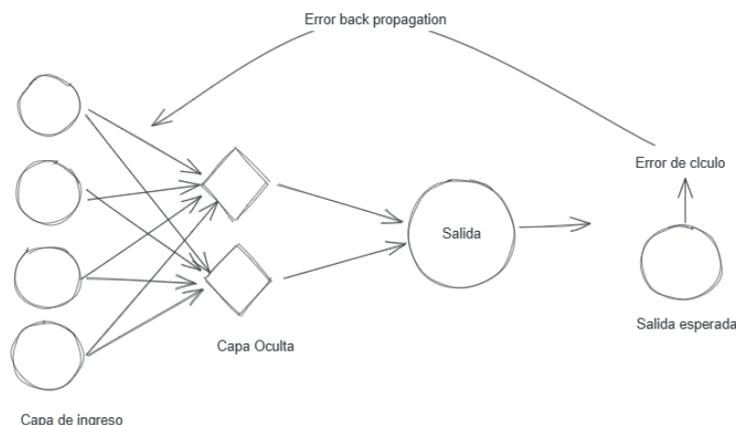


Fig. 1. Descripción gráfica de la propagación hacia atrás

1.5.7. Redes neuronales convolucionales (CNN):

Las redes convolucionales son una variante particular de las redes neuronales diseñadas para trabajar con datos que tienen una estructura espacial, como imágenes o señales de audio. se componen de tres tipos de capas: capas convolucionales que extraen características, capas de agrupamiento que reducen la dimensionalidad, y capas totalmente conectadas que establecen las conexiones finales[37].

Las redes convolucionales utilizan capas convolucionales para extraer características locales de los datos de entrada, seguidas de capas de agrupación para reducir la dimensión de los datos. Las convoluciones se pueden representar de la siguiente manera:

$$S(i, j) = (I * K)(i, j) = \sum_m \sum_n I(m, n)K(i - m, j - n)$$

donde S(i,j) es el valor del píxel en la posición (i,j) de la imagen resultante, I es la imagen de entrada, K es el kernel de convolución y * denota la operación de convolución.

Otra fórmula importante es la función de activación ReLU (unidad lineal rectificadora), que se define como:

$$ReLU(x) = \max(0, x)$$

La función softmax es también muy común en las redes convolucionales, y se define como:

$$softmax(z_i) = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}}$$

Donde Z_i es la entrada de la neurona i y K es el número total de neuronas en la capa.

Además, la función de pérdida de entropía cruzada se utiliza a menudo como una medida de la diferencia entre la salida de la red y la salida esperada. La fórmula de la entropía cruzada se puede expresar como:

$$H(p, q) = - \sum_x p(x) \log(q(x))$$

Donde p es la distribución real de los datos y q es la distribución predicha por la red.

Otras fórmulas importantes en las redes convolucionales incluyen las operaciones de pooling (como max-pooling y average-pooling) y las funciones de regularización (como la regularización L1 y L2).

1.5.8. Capa Max Pooling:

La capa de max pooling es comúnmente utilizada en las redes neuronales convolucionales para reducir el tamaño de la salida de las capas convolucionales, disminuyendo así el total de parámetros y, por lo tanto, el tiempo se toma para el entrenamiento. Esta capa funciona tomando el valor máximo de un conjunto de activaciones vecinas en una ventana deslizante Según [38], la capa de max pooling

es una forma de realizar una operación de submuestreo que reduce la resolución espacial de la entrada. Esta capa es efectiva en la detección de características de nivel superior y permite a la red convolucional ser más invariante a pequeñas traslaciones en la entrada.

$$y_{i,j,k} = \max_{u,v} x_{(i-1)S+u,(j-1)S+v,k}$$

Donde:

- I. x es la matriz de entrada de la capa convolucionales.
- II. y es la matriz resultante después de aplicar la operación de Max Pooling.
- III. i, j, y k son índices que representan la posición de un elemento en la matriz de salida.
- IV. u y v son índices que representan la posición de un elemento en una submatriz de la matriz de entrada.
- V. S es el valor de stride, que determina el tamaño del paso que se da al mover la submatriz sobre la matriz de entrada.

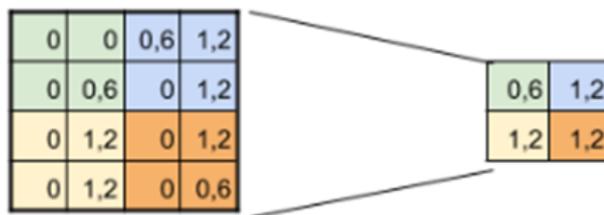


Fig. 2. Submuestreo cromático de 2x2 reduciendo a la mitad

1.5.9. Capa de clasificación

Después de múltiples capas de convolución y agrupación máxima, se utiliza un perceptrón multicapa superficial (MLP) para completar la MPCNN. La capa de salida tiene una neurona por capa en el problema de clasificación [39]

$$y = \text{softmax}(Wx + b)$$

1.5.10. Python

Python es un lenguaje popular que es categorizado como lenguaje de programación de nivel superior ampliamente empleado en numerosos proyectos de aprendizaje profundo debido a su simplicidad y a su ecosistema de bibliotecas especializadas [40].

Las bibliotecas como TensorFlow y PyTorch, desarrolladas por Google y Facebook respectivamente, han impulsado el uso de Python en el campo del aprendizaje profundo [39].

1.5.11. PyTorch

PyTorch es una biblioteca de aprendizaje automático diseñada para admitir estilos de programación imperativo y Python, admitir código como modelo, simplificar la depuración y ser compatible con otras bibliotecas informáticas científicas populares, sin dejar de ser potente y compatible con aceleradores de hardware como GPU [41]. Diseñado para la conveniencia y la velocidad, es un programa regular de Python con control total del usuario [41]. PyTorch se usa a menudo junto con otras bibliotecas y marcos de trabajo de Python, como NumPy, SciPy y OpenCV [42]. También se utiliza para tareas como el cálculo de núcleos tangentes empíricos (NTK) para modelos de redes neuronales [43] y normalización de flujo [44]. PyTorch se basa en el popular marco de aprendizaje profundo PyTorch, que permite una fácil integración de PyTorch en modelos más grandes o procesos de aprendizaje automático [44].

1.5.12. Labellmg

Software de anotación/etiquetado/etiquetado de datos para procesos de aprendizaje automático. Está disponible en GitHub y se puede utilizar para anotar datos y extraer conjuntos de datos de VOC [45], [46]. A menudo se usa junto con otras bibliotecas y marcos de Python como Bokeh, HoloViews, NumPy, OpenCV y TensorFlow [43] [47]. Este software permite al usuario dibujar límites alrededor de los objetos en la imagen, que luego se pueden usar para entrenar modelos de aprendizaje automático [43].

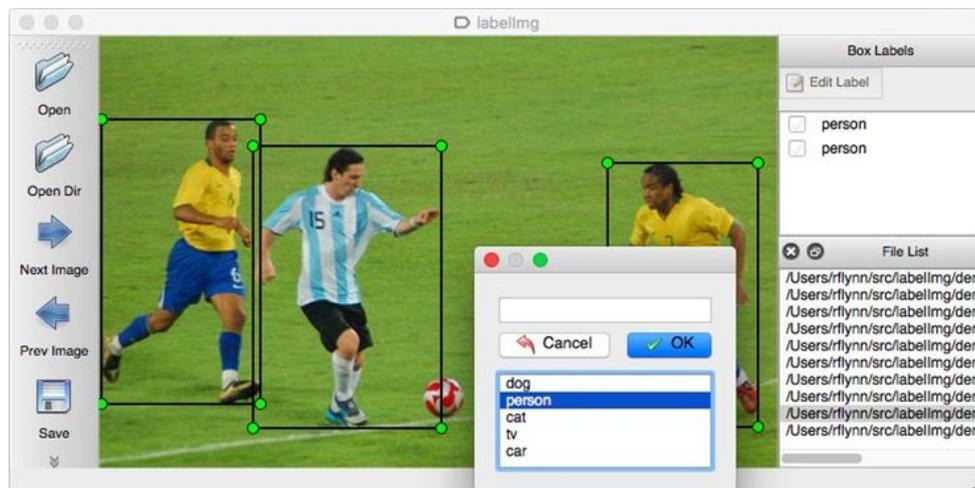


Fig. 3. Labellmg para la tarea de anotación

1.5.13. Tensorflow

Es una biblioteca que ejecuta algoritmos de aprendizaje automático con JavaScript, dentro del entorno de un navegador y/o en el servidor, este ha permitido al grupo de desarrolladores de JavaScript poder construir modelos de aprendizaje continuo [42].

Según [43], TensorFlow es una biblioteca de aprendizaje profundo desarrollada por Google que permite a los usuarios crear, entrenar e implementar modelos de aprendizaje profundo, incluidas las redes neuronales convolucionales,

para diversas aplicaciones de visión por computadora. Con la ayuda de TensorFlow, los usuarios pueden crear y entrenar modelos de CNN desde cero o utilizar modelos pre-entrenados para resolver problemas específicos de clasificación de imágenes.

Muchos de los modelos más usados para la detección de movimiento en tiempo real son EfficientDet, YOLO, MobileNet, etc

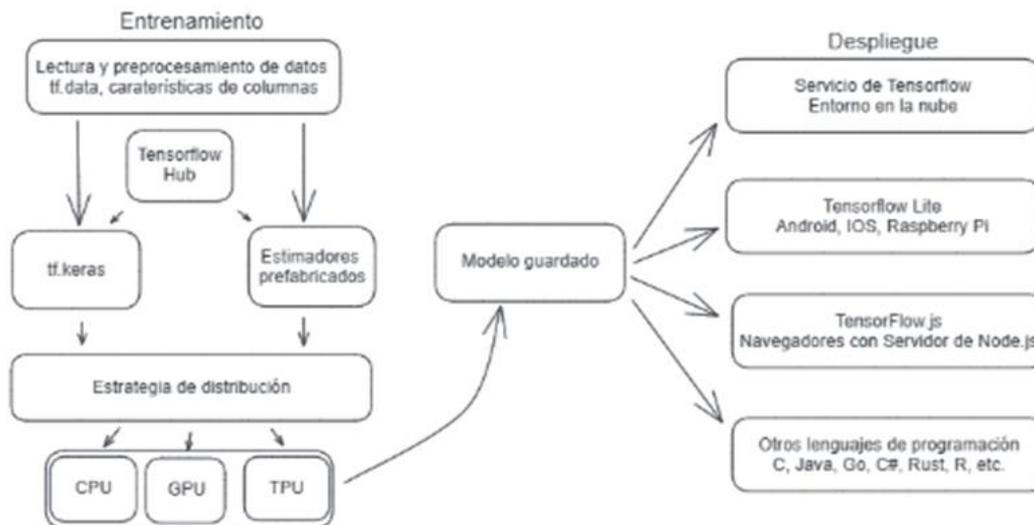


Fig. 4. Diagrama de procesamiento de Tensorflow

1.5.14. Tensorflow.js

TensorFlow.js es un marco de trabajo de código abierto creado por Google. que permite el entrenamiento y la ejecución de modelos de aprendizaje automático directamente en navegadores web y entornos basados en Node.js. Proporciona una interfaz de alto nivel basada en JavaScript para construir y desplegar modelos de aprendizaje automático, lo que permite aprovechar la potencia de la GPU del cliente para realizar cálculos intensivos y acelerar el procesamiento de los modelos.[44]

TensorFlow.js tiene como objetivo principal habilitar la implementación de inteligencia artificial en aplicaciones web sin depender de servicios en la nube. Permite realizar tareas de clasificación de imágenes u objetos para automático,

directamente en el navegador del usuario.

Una de las ventajas clave de TensorFlow.js es su capacidad para aprovechar la GPU del cliente, lo que permite un procesamiento más rápido y eficiente de los modelos de aprendizaje automático. Esto significa que se pueden ejecutar tareas complejas en tiempo real sin requerir una conexión constante a servicios remotos.

1.5.15. WebGL

WebGL (Web Graphics Library) es una API de gráficos en 3D basada en OpenGL ES que permite la ejecución eficiente de gráficos acelerados por hardware en navegadores web modernos. Permite a los desarrolladores web utilizar el poder de la GPU (Unidad de Procesamiento Gráfico) para realizar cálculos y renderizar gráficos en tiempo real dentro de un entorno web.

En el contexto de Tensorflow.js, WebGL se utiliza como un backend de aceleración para ejecutar operaciones matemáticas y de tensor en paralelo en la GPU. Esto permite un rendimiento significativamente más rápido en comparación con la ejecución en la CPU, especialmente para tareas intensivas en cálculos como entrenamiento y predicción de modelos de aprendizaje automático [45].

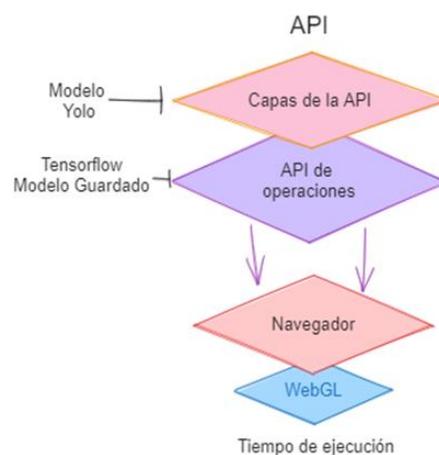


Fig. 5. Uso de Tensorflow con WebGL.

1.5.16. Sólo se mira una vez (YOLO)

Sólo se mira una vez o You Only Look Once (YOLO) en inglés es un modelo de detección de objetos en tiempo real que utiliza una sola pasada por la red neuronal convolucional para predecir las regiones de interés y las clases de objetos en una imagen [46]. A diferencia de otros enfoques de detección que emplean algoritmos de dos etapas, YOLO realiza la detección y clasificación directamente en una sola etapa.

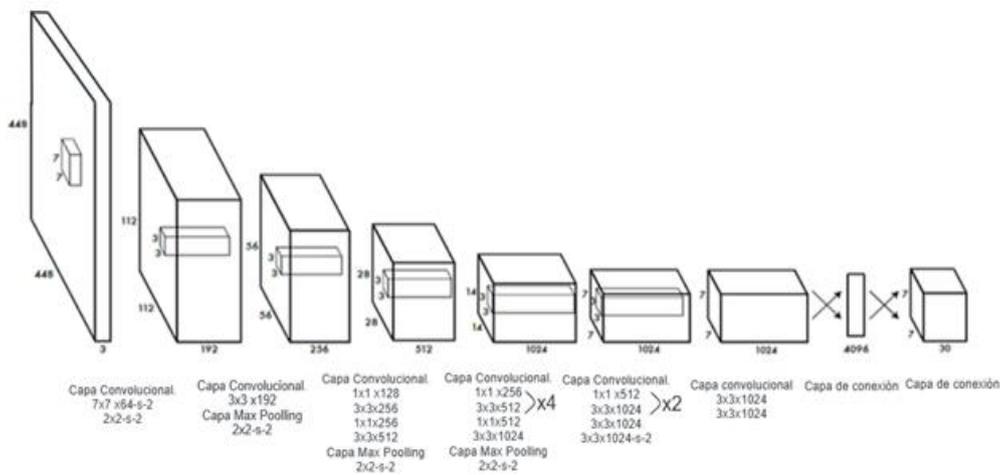


Fig. 6. Arquitectura You Only Look Once (YOLO).

1.5.17. WebSockets

Los WebSockets son un protocolo de red basado en TCP que permite la comunicación bidireccional y en tiempo real entre un cliente y un servidor. Los WebSockets establecen una conexión persistente entre el cliente y el servidor, lo que permite que los datos se transmitan de manera eficiente y en tiempo real. Este estándar es comúnmente empleado en aplicaciones web que necesitan una comunicación instantánea, como juegos en línea, sistemas de chat y servicios de streaming de video.[48]

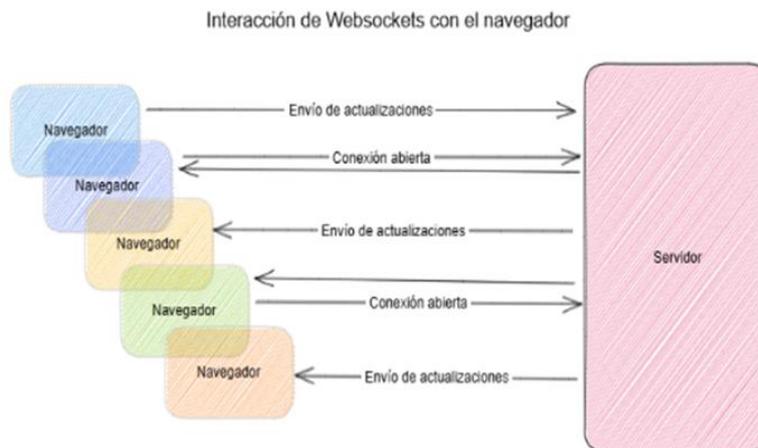


Fig. 7. Interacción WEBSOCKET

1.5.18. Comunicación Punto a Punto (P2P)

En este tipo de comunicación, cada dispositivo actúa como un cliente y un servidor al mismo tiempo, lo que permite una comunicación más rápida y eficiente. En el contexto de las videollamadas, la comunicación peer to peer permite que los usuarios se comuniquen directamente entre sí sin la necesidad de que los datos pasen por un servidor centralizado, lo que puede mejorar la calidad de la llamada y reducir la latencia. Además, la comunicación peer to peer o punto a punto puede ser más segura que la comunicación a través de un servidor centralizado, ya que no hay un punto central de fallo o vulnerabilidad. En lugar de eso, los dispositivos se conectan directamente entre sí utilizando una dirección IP única para cada dispositivo. Esto permite que los datos se transmitan de manera más rápida y eficiente, lo que puede mejorar la calidad de la llamada y reducir la latencia.[49]

1.5.19. Web RTC

Las conexiones directas entre pares o peer-to-peer pueden ofrecer una latencia mínima, eso hace que los juegos alimentación de sensores, parezcan más rápidos en tiempo real, WebRTC fue creado para poder transmitir videos en tiempo real para que parezca más rápidos e interactivos o en tiempo real. Es un proyecto

abierto y gratuito que proporciona a los navegadores la capacidad de poder realizar comunicación en tiempo real a través de un api sencillo (RTC) [50]

WebRTC es una tecnología de comunicación en tiempo real que permite el envío de audio, video y datos a través de la web sin necesidad de plugins externos [51] Es una API gratuita y de código abierto desarrollada por Google que utiliza protocolos y tecnologías como el Real-time Transport Protocol (RTP), el Session Description Protocol (SDP) y el Interactive Connectivity Establishment (ICE) para establecer conexiones entre navegadores web [49].

1.5.20. Node.js

Node.js es un entorno de ejecución de JavaScript multiplataforma y de código abierto que se utiliza para ejecutar aplicaciones web en el servidor [52]. Node.js se basa en el motor JavaScript V8 de Chrome y permite a los desarrolladores escribir scripts del lado del servidor utilizando JavaScript. Node.js es conocido por su capacidad para manejar un gran número de conexiones concurrentes con gran eficiencia y escalabilidad. Además, Node.js emplea un enfoque de programación asíncrona y no bloqueante, permitiéndole manejar múltiples solicitudes de manera simultánea sin detener el proceso principal. Esta característica lo convierte en una opción ideal para aplicaciones web en tiempo real, como juegos en línea, servicios de chat y plataformas de streaming de video.



Fig. 8. Arquitectura de Node.js del uso de hilos.

1.5.21. Express.js

Express.js es un framework minimalista y versátil diseñado para el desarrollo de aplicaciones web en Node.js. Con Express.js, puedes crear rápidamente servidores web eficientes y definir fácilmente rutas para manejar las solicitudes HTTP entrantes. Además, Express.js ofrece la posibilidad de configurar enrutadores modulares, lo que facilita la organización de las rutas y los controladores en diferentes módulos o archivos. Esto mejora la estructura y la legibilidad del código, especialmente en aplicaciones web más grandes [53].

1.5.22. React.js

Es una biblioteca basada en el lenguaje de programación JavaScript que se usa para poder implementar interfaces de usuario reutilizables, Además permite el desarrollo de grandes aplicaciones que hace que puedan ser dinámicos sin necesidad de refrescar una página eso ayuda al performance [54].

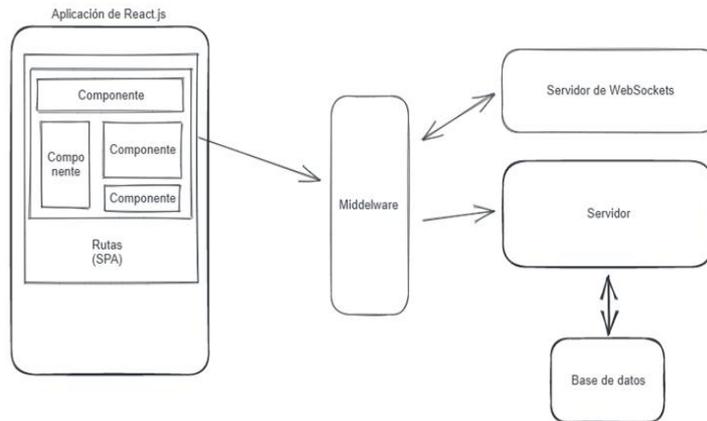


Fig. 9. Diagrama de uso de React.js y conexión con servicios.

II. MATERIAL Y MÉTODO

2.1 Tipo y Diseño de Investigación

El tipo de investigación es cuantitativo debido a que se tomó el criterio científico con el cual está basado en el estudio computacional. Con esto, los datos adquiridos dentro del estudio fueron procesados para un análisis estadístico.

Esta investigación es un estudio cuasi experimental puesto que tiene como objetivo probar una hipótesis identificando si el grupo de comparación es lo más similar al grupo de tratamiento en términos de las características iniciales de la investigación ya sean por razones logísticas o éticas. [55].

2.2 Variables, Operacionalización

TABLA 1. VARIABLES, OPERACIONALIZACIÓN

Variable de estudio	Definición conceptual	Definición operacional	Dimensiones	Indicadores	Ítems	Instrumento	Valores finales	Tipo de variable	Escala de medición
Modelo de Traducciones de lenguaje de señas	Un modelo de IA con el fin aprender de datos y realizar tareas sofisticadas	$\frac{1}{n} \sum_{k=1}^{k=n} AP_K$	Velocidad de detección	Valor medio de precisión media (mAP)	Eficiencia de respuesta del modelo	Registro electrónico	Valores decimales	Numérico	Bajo: mayor a 60% Medio: mayor a 80% Alto: mayor a 90%
	Estima el estándar y determinando la medición del modelo	$\frac{TP}{TP + FP}$	Capacidad de detección	Precisión(P)	Eficiencia de respuesta del modelo	Registro electrónico	valores decimales	Numérico	Bajo: mayor a 0.6 Medio: mayor a 0.8 Alto: mayor a 0.9
	Gráfica que muestra la relación entre la confianza (umbral de confianza) utilizado en la clasificación de un modelo y el valor de F1-Score	$\frac{2 * P * R}{P + R}$	Umbral de confianza	F1-Confidence Curva	F1-Score	Registro electrónico	valores decimales	Numérico	Bajo: mayor a 0.6 Medio: mayor a 0.8 Alto: mayor a 0.9
	Evalúa la eficacia de los modelos de clasificación	$\frac{TP}{TP + FN}$	Velocidad de detección	Recuperación (Re)	Eficiencia de respuesta del modelo	Registro electrónico	valores decimales	Numérico	Bajo: mayor a 0.4 Medio: mayor a 0.7 Alto: mayor a 0.9

Evalúa el desempeño del modelo y hacer mejoras para alcanzar el objetivo de los modelos	$TT = TPP + TI + TNMS$	Velocidad de detección	Rendimiento (Rn)	Eficiencia de respuesta del modelo	Registro electrónico	valores decimales	Numérico	Bajo: mayor a 0.4 Medio: mayor a 0.7 Alto: mayor a 0.9
Calcula el tiempo que tarda un proceso en completarse	$\frac{\sum td}{n}$	La precisión en el modelo de aprendizaje	Tiempo de procesamiento	Eficiencia de respuesta del modelo en la aplicación web	Registro electrónico	valores decimales	Numérico	Bajo: alrededor de 800 ms Medio: alrededor de 600 ms Alto: alrededor de 200 ms
Implementación del modelo de aprendizaje e profundo	$\frac{p2 - p1}{t2 - t1} * 100$ <i>Contador de procesador</i>	Consumo promedio del CPU	Porcentaje de uso del CPU	Eficiencia de respuesta de la aplicación web	Registro electrónico	Valores decimales	Numérico	Bajo: mayor a 0.6 Medio: mayor a 0.8 Alto: mayor a 0.9
Escala de satisfacción (ES)	1 – <i>Precisión</i>	Eficiencia	Tasa de errores	Eficiencia de comunicación	Registro electrónico	Valores numéricos	Numérico	Bajo: Entre 1 y 2 Medio: Entre 3 y 4 Alto: 5

2.3 Población de estudio, muestra, muestreo y criterios de selección

2.3.1 Población

La población está dada por modelos de aprendizaje profundo los cuales se listan en el anexo 4

2.3.2 Muestra

La muestra fue determinada por conveniencia y debido a los antecedentes encontrados de los modelos de aprendizaje profundo los cuales son: YOLO y Mobile Net V2.

2.4 Técnicas e instrumentos de recolección de datos, validez y confiabilidad

Los instrumentos que son de apoyo en esta investigación son las siguientes

Registro electrónico

Es un instrumento que se usó para analizar los resultados que se tiene con la precisión, rendimiento, recuperación y mAP. Además, se utilizará la consola de navegador para realizar una monitorización, y así medir el consumo de tareas del navegador, mostrar registros de uso, ingresos y procesos del servidor.

2.5 Procedimiento de análisis de datos

a. Tiempo de procesamiento

Tiempo que tarda el servidor en responder a la solicitud de un determinado usuario a través del navegador.

$$TRP = \frac{\sum td}{n}$$

TRP = Tiempo de respuesta promedio

td = Tiempo de duración de la transacción

n = Número de transacciones iniciadas

b. Consumo promedio del CPU

Es una métrica que ayuda a calcular la utilización de la CPU en el navegador utilizando la siguiente fórmula:

$$CPU \% = \frac{\frac{p2 - p1}{t2 - t1} * 100}{\text{Contador de procesador}}$$

p1 = Porcentaje de tiempo de procesador al ejecutar por primera vez la consulta

p2 = Porcentaje de tiempo de procesador por segunda vez al ejecutar la consulta

t2 y t1 = Marca de tiempo Sys 100NS

c. Tasa de Error:

La métrica de Tasa de Error medirá el rendimiento en la traducción del lenguaje de señas, indicando el porcentaje de predicciones erróneas en comparación con el total de detecciones efectuadas.

$$TE = 1 - P$$

TE = Tasa de Error

P = Precisión del modelo de clasificación

d. Valor medio de precisión media (mAP):

Métrica que mide el rendimiento del modelo para detectar los objetos, la recuperación de las imágenes y detecta los gestos asociados con las señas del lenguaje de señas.

$$mAP = \frac{1}{n} \sum_{k=1}^{k=n} AP_K$$

n = número de clases

AP_K = Precisión Promedio Media en K

e. F1-Confidence Curva

Esta métrica es una combinación de precisión y recuerdo, ayuda a determinar la precisión y la sensibilidad promedio ponderada de las traducciones del lenguaje de señas.

$$2 * \frac{P * R}{P + R}$$

P = precisión del modelo de clasificación

R = Recuperación

f. Precisión (P):

La precisión que se determinará estará relacionada con la proporción de observaciones positivas correctamente con respecto al número total pronosticado de observaciones positivas vistas en la interpretación del lenguaje de señas.

$$P = \frac{TP}{TP + FP}$$

TP = Verdadero positivo

FP = Falso Positivo

g. Recuperación (Re):

La métrica de recuperación es la relación entre las observaciones positivas pronosticadas correctamente y todas las observaciones en tiempo real durante la interpretación del lenguaje de señas.

$$R = \frac{TP}{TP + FN}$$

TP = Verdadero positivo

FN = Falso negativo

h. Rendimiento (Rn):

La métrica de rendimiento calcula el tiempo total necesario para procesar la detección de objetos en una imagen, obteniendo así el tiempo total de ejecución del proceso.

$$TT = TPP + TI + TNMS$$

TT: Tiempo Total.

TPP: Tiempo de Preprocesamiento.

TI: Tiempo de Inferencia.

TNMS: Tiempo de NMS (Supresión No Máxima).

2.6 Criterios éticos

Se utilizó el principio ético de la propiedad intelectual, debido a que la información extraída la cual nos ayudó para poder tener un mayor conocimiento para darle el desarrollo que se necesita para el presente proyecto se citó y se realizó la respectiva referencia a los trabajos de investigación, papers y libros.

III. RESULTADOS Y DISCUSIÓN

3.1. Resultados

En esta sección, se presentarán y explicarán los resultados del sistema propuesto para el reconocimiento del alfabeto del lenguaje de señas. Se utilizó un conjunto de datos compuesto por un total de 3,596 imágenes del alfabeto de lenguaje de señas, las cuales fueron recopiladas y organizadas de diversas personas en diferentes condiciones climáticas. Esta amplia diversidad permitió medir y evaluar exhaustivamente el desempeño del sistema.

Variable Independiente

En este estudio, se emplearon tres modelos para la detección de objetos: YOLO v5, YOLO v5 Personalizado y MobileNet v2. Dichos modelos fueron entrenados y evaluados utilizando un conjunto de datos estándar, con el objetivo de determinar su eficacia en el reconocimiento del lenguaje de señas mediante la detección de gestos correspondientes a las letras del abecedario.

A continuación, se presenta un resumen de los resultados:

- **Valor medio de precisión media (mAP)**

TABLA II. RESULTADOS DE VALOR MEDIO DE PRECISIÓN MEDIA

Modelo	mAP
YOLO V5	0.902
YOLO V5 personalizado	0.881
MobileNet V2	0.727

Los resultados muestran que el modelo YOLO v5 alcanzó el mayor valor medio de precisión media con un valor de 0.902, seguido de cerca por el modelo YOLO v5 personalizado con una mAP de 0.881. Por otro lado, MobileNet v2 obtuvo un valor medio de precisión media de 0.727 en la tarea de reconocimiento del lenguaje de señas.

La Tabla II proporciona una visión general de la capacidad de los modelos para realizar la identificación de gestos característicos del lenguaje de señas. Los valores más altos de mAP indican una mayor capacidad de los modelos para reconocer con precisión los elementos clave del lenguaje de señas en las imágenes proporcionadas.

A continuación, se presenta un gráfico de barras que representa visualmente los resultados de mAP de los modelos:

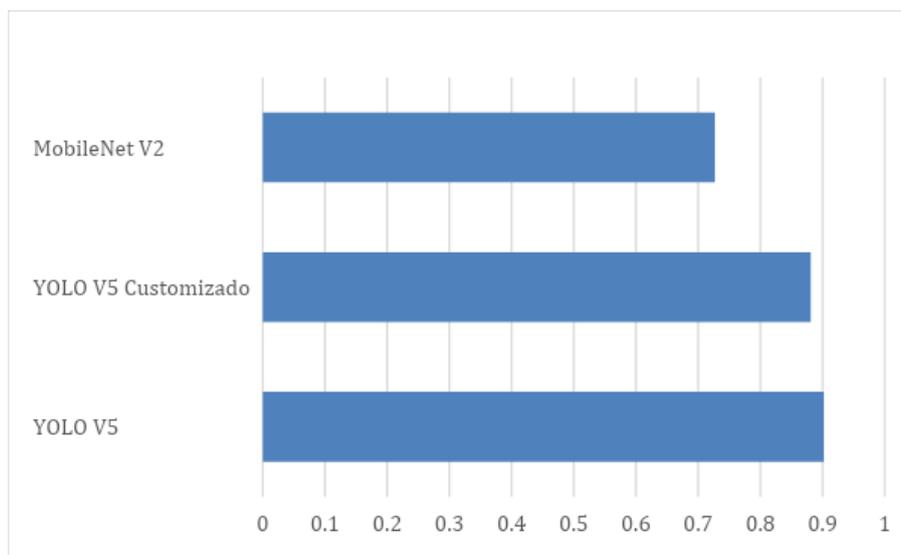


Fig. 10. Resultados de mAP de los modelos implementados

- **Precisión**

TABLA III. RESULTADOS DE PRECISIÓN.

Modelo	Precisión
YOLO V5	0.976
YOLO V5 personalizado	0.972
MobileNet V2	0.787

Nota: comparación de resultados de precisión de los modelos implementados

Se evaluaron tres modelos: YOLO v5, YOLO v5 personalizado y MobileNet v2. Por lo que YOLO v5 demostró la mayor precisión con un valor de 0.976, seguido de cerca por YOLO v5 personalizado con 0.972. Estos resultados indican que aproximadamente el 97.6% y

97.2% de las predicciones, respectivamente, fueron correctas en la identificación de gestos y signos del lenguaje de señas. Por otro lado, MobileNet v2 obtuvo una precisión de 0.787, lo que representa un 78.7% de precisión en sus predicciones. En resumen, los modelos YOLO v5 y YOLO v5 personalizado destacan por su alta precisión en el reconocimiento del lenguaje de señas, superando significativamente a MobileNet v2 en este aspecto.

A continuación, se muestran las visualizaciones de precisión de los modelos YOLO v5, YOLO v5 personalizado y MobileNet v2 en el reconocimiento del lenguaje de señas:

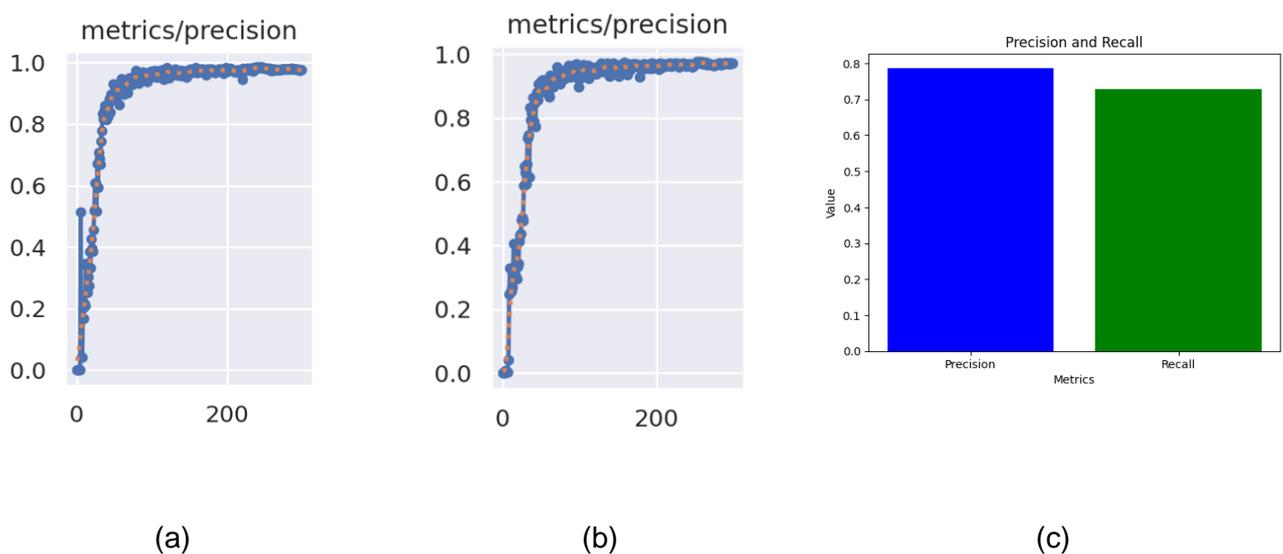


Fig. 11. (a) Representa la matriz de precisión del modelo YOLO v5 (b) Representa la precisión del modelo YOLO v5 personalizado (c) Gráfico de barras que representa visualmente la precisión del modelo MobileNet v2.

- **Recuperación**

TABLA IV. RESULTADOS DE RECUPERACIÓN.

Modelo	Recuperación
YOLO v5	0.820
YOLO V5 personalizado	0.808
Mobile Net V2	0.727

Nota: comparación de resultados de recuperación de los modelos implementados

Los resultados de recuperación (recall) muestran que el modelo YOLO v5 tiene una tasa de 0.820, lo que indica que puede identificar correctamente aproximadamente el 82% de los casos positivos en el conjunto de datos. Por otro lado, el modelo YOLO v5 personalizado tiene una tasa de 0.808, ligeramente más baja que el YOLO v5 estándar. Esto sugiere que el segundo modelo puede estar perdiendo la detección de aproximadamente el 19.2% de los casos positivos que el YOLO v5 logra capturar. En comparación, el modelo MobileNet V2 tiene una tasa de recuperación de 0.727, lo que indica que puede identificar correctamente aproximadamente el 72.7% de los casos positivos en el conjunto de datos. Este valor más bajo señala que hay una mayor proporción de casos positivos que no se están detectando correctamente en comparación con los otros dos modelos.

A continuación, se muestran las visualizaciones de recuperación de los modelos YOLO v5, YOLO v5 personalizado y MobileNet v2 en el reconocimiento del lenguaje de señas:

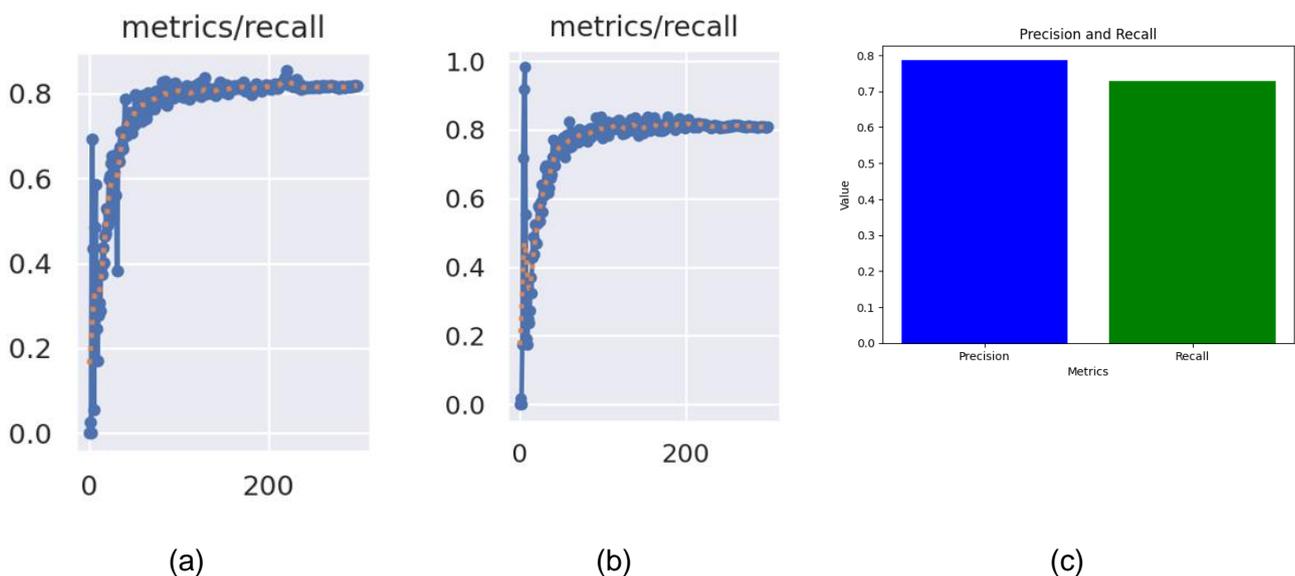


Fig. 12. (a) Representa la matriz de recuperación del modelo YOLO v5 (b) Representa la recuperación del modelo YOLO v5 personalizado (c) Gráfico de barras que representa visualmente la recuperación del modelo MobileNet v2.

- **F1-Confidence Curva**

Los valores de F1-Confidence representan el equilibrio entre precisión y recuperación para cada modelo en los respectivos puntos de confianza dados en los anteriores datos.

TABLA V. RESULTADOS DE CÁLCULO DE LA SENSIBILIDAD F1-CONFIDENCE.

Modelo	F1-Confidence
YOLO	0.8918
YOLO V5 personalizado	0.8818
Mobile Net V2	0.7552

Nota: comparación de resultados de sensibilidad de los modelos implementados

En la evaluación de los modelos YOLO, YOLO V5 personalizado y MobileNet V2 para detectar objetos, se observa que el modelo YOLO tiene el mejor desempeño con un puntaje de aproximadamente 0.8918 en F1-Confidence. Le sigue el YOLO V5 Personalizado con un puntaje cercano de 0.8818, mostrando también una buena capacidad para detectar objetos. En contraste, el MobileNet V2 obtiene alrededor de 0.7552 en F1-Confidence, lo que indica una precisión un poco menor en la detección de objetos. En resumen, estos resultados muestran que el YOLO es el más efectivo para detectar objetos, seguido por el YOLO V5 Personalizado y, por último, el MobileNet V2.

A continuación, se presenta un gráfico de barras que representa visualmente los resultados de F1-Confidence de los modelos

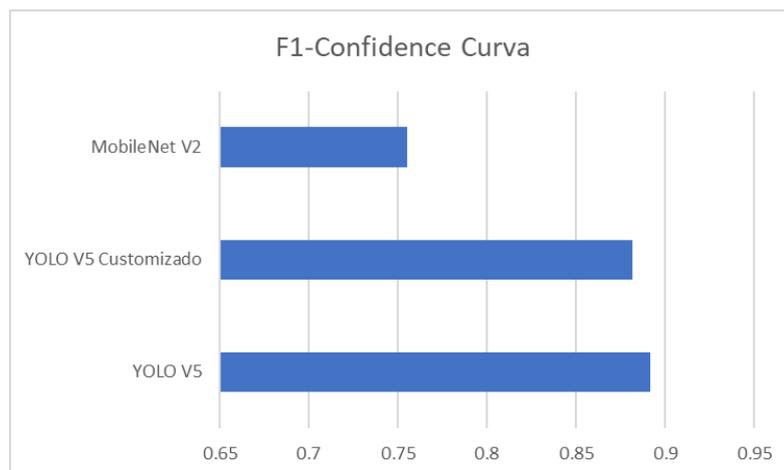


Fig. 13. Resultados de F1-Confidence Curva de los modelos implementados.

- Rendimiento

La velocidad de detección del modelo se ha medido para un conjunto total de 1196 imágenes que fueron adquiridas y analizadas durante el proceso de validación.

Modelo YOLO v5 personalizado

```

lls
python detect.py --weights runs/train/yolov5s_results2/weights/best.pt --img 256 --conf 0.4 --source /content/drive/MyDrive/dataset-completed/test/images
image 1141/1196 /content/drive/MyDrive/dataset-completed/test/images/u.c4004395-300f-474f-870e-20398telc0aea.jpg: 256x256 1 u, 0.7ms
image 1142/1196 /content/drive/MyDrive/dataset-completed/test/images/u.c522ad12-5960-4b22-88e9-72e84cea6bda.jpg: 256x256 1 u, 6.6ms
image 1143/1196 /content/drive/MyDrive/dataset-completed/test/images/u.c8019c66-aa4b-45bf-9158-c65ca78fa279.jpg: 256x256 1 u, 6.6ms
image 1144/1196 /content/drive/MyDrive/dataset-completed/test/images/u.cb16c831-22b3-4756-aa9d-48ab8c268d66.jpg: 256x256 1 u, 7.6ms
image 1145/1196 /content/drive/MyDrive/dataset-completed/test/images/u.da95f6f1-61b7-499a-b442-9ec931e94715.jpg: 256x256 1 u, 8.3ms
image 1146/1196 /content/drive/MyDrive/dataset-completed/test/images/u.dde6070e-6d2e-4e11-8eb3-b9aa2322847e.jpg: 256x256 1 u, 6.6ms
image 1147/1196 /content/drive/MyDrive/dataset-completed/test/images/u.e3b1ae75-e424-4fab-b197-a5e3ac5e638d.jpg: 256x256 1 u, 6.6ms
image 1148/1196 /content/drive/MyDrive/dataset-completed/test/images/u.fcf27031-e429-4189-aeda-6704fa417797.jpg: 256x256 1 u, 7.7ms
image 1149/1196 /content/drive/MyDrive/dataset-completed/test/images/v.d72ac46d-73ad-45ee-a90d-1a6eead67ffe.jpg: 256x256 1 v, 6.7ms
image 1150/1196 /content/drive/MyDrive/dataset-completed/test/images/v.d9469b19-cfc9-4d36-96ab-3450db287efe.jpg: 256x256 1 v, 8.3ms
image 1151/1196 /content/drive/MyDrive/dataset-completed/test/images/v.d969b781-1e03-4ff2-a53d-06443e0fff55.jpg: 256x256 1 v, 7.5ms
image 1152/1196 /content/drive/MyDrive/dataset-completed/test/images/v.da474b94-fb3a-4c44-b295-588bd4424272.jpg: 256x256 1 v, 6.7ms
image 1153/1196 /content/drive/MyDrive/dataset-completed/test/images/v.da4ac18a-1743-4425-82d1-0134197a0465.jpg: 256x256 1 v, 6.6ms
image 1154/1196 /content/drive/MyDrive/dataset-completed/test/images/v.e0cd47c5-3abd-45de-8942-f5b6b204be01.jpg: 256x256 1 v, 7.6ms
image 1155/1196 /content/drive/MyDrive/dataset-completed/test/images/v.e4644c2f-4f26-4cc9-904f-f06a4055ce7c.jpg: 256x256 1 v, 6.4ms
image 1156/1196 /content/drive/MyDrive/dataset-completed/test/images/v.e52128cd-0474-4656-8c8d-79cf09f948e5.jpg: 256x256 1 v, 7.6ms
image 1157/1196 /content/drive/MyDrive/dataset-completed/test/images/v.e6bb7f1e-147b-421b-8ae8-08745380bd4f.jpg: 256x256 1 v, 6.5ms
image 1158/1196 /content/drive/MyDrive/dataset-completed/test/images/v.ee55f273-d3ca-4636-863c-81f7bd7f848.jpg: 256x256 1 v, 6.6ms
image 1159/1196 /content/drive/MyDrive/dataset-completed/test/images/v.f8ce6277-c98c-4edc-802d-085f2ecc238f.jpg: 256x256 1 v, 6.7ms
image 1160/1196 /content/drive/MyDrive/dataset-completed/test/images/v.fafb54fc-e8d2-42fc-a36a-b3dc546bd968.jpg: 256x256 1 v, 6.8ms
image 1161/1196 /content/drive/MyDrive/dataset-completed/test/images/w.d279f981-2a2c-4397-b3e9-469dedd26e0e.jpg: 256x256 1 w, 9.5ms
image 1162/1196 /content/drive/MyDrive/dataset-completed/test/images/w.d574b29b-2256-40f9-9046-faa63a882b52.jpg: 256x256 1 w, 7.3ms
image 1163/1196 /content/drive/MyDrive/dataset-completed/test/images/w.d8de226a-573f-452c-1b11-f232c3c44682.jpg: 256x256 1 w, 6.5ms
image 1164/1196 /content/drive/MyDrive/dataset-completed/test/images/w.d95fee7c-85d0-4707-b936-0c7db108a511.jpg: 256x256 1 w, 8.8ms
image 1165/1196 /content/drive/MyDrive/dataset-completed/test/images/w.dbcc4aac-8625-4ecc-9d59-c0db26310352.jpg: 256x256 1 w, 6.6ms
image 1166/1196 /content/drive/MyDrive/dataset-completed/test/images/w.df0c6dcb-25ac-491e-8cc7-bf28813f1051.jpg: 256x256 1 w, 6.6ms
image 1167/1196 /content/drive/MyDrive/dataset-completed/test/images/w.dff1ac58-66f1-48ba-a1dc-53d954930224.jpg: 256x256 1 w, 6.5ms
image 1168/1196 /content/drive/MyDrive/dataset-completed/test/images/w.e10f5701-7aa7-45db-858c-632ef273733a.jpg: 256x256 1 w, 6.4ms
image 1169/1196 /content/drive/MyDrive/dataset-completed/test/images/w.e1cc2359-ba67-4865-89f8-2ada344a39f5.jpg: 256x256 1 w, 6.7ms
image 1170/1196 /content/drive/MyDrive/dataset-completed/test/images/w.e57d6009-02c7-4e03-b0c7-b28b6f98cb6b.jpg: 256x256 1 w, 6.6ms
image 1171/1196 /content/drive/MyDrive/dataset-completed/test/images/w.e8ece4f2-3ea8-413a-b9ac-96d8f0d41917.jpg: 256x256 1 w, 6.6ms
image 1172/1196 /content/drive/MyDrive/dataset-completed/test/images/w.efeb9c7c-e2ff-4cab-b638-30fad4ffeaca.jpg: 256x256 1 w, 7.4ms
image 1173/1196 /content/drive/MyDrive/dataset-completed/test/images/x.142782f7-96bc-4fb5-acc6-c900bd1a139.jpg: 256x256 1 x, 7.8ms
image 1174/1196 /content/drive/MyDrive/dataset-completed/test/images/x.267252ec-d0ed-4f3c-a1d0-901f53321d2f.jpg: 256x256 1 x, 6.5ms
image 1175/1196 /content/drive/MyDrive/dataset-completed/test/images/x.94897b9a-1906-4d50-b451-9e42854dbdfd.jpg: 256x256 1 x, 6.8ms
image 1176/1196 /content/drive/MyDrive/dataset-completed/test/images/x.ab29ea08-8965-4a84-ae0a-ab50eab76849.jpg: 256x256 1 x, 6.5ms
image 1177/1196 /content/drive/MyDrive/dataset-completed/test/images/x.b05f2637-6f24-47c4-ac92-d94f3b56ff6a.jpg: 256x256 1 x, 6.6ms
image 1178/1196 /content/drive/MyDrive/dataset-completed/test/images/x.b7a10f2a-6df7-4f64-a76b-6048888d0a28.jpg: 256x256 1 x, 9.8ms
image 1179/1196 /content/drive/MyDrive/dataset-completed/test/images/x.c04b702e-82a2-408f-92b8-5e696d5f9884.jpg: 256x256 1 x, 6.9ms
image 1180/1196 /content/drive/MyDrive/dataset-completed/test/images/x.c69fc46f-85fb-444f-90ab-a82f981b37f4.jpg: 256x256 1 x, 11.6ms
image 1181/1196 /content/drive/MyDrive/dataset-completed/test/images/x.c71ed449-9d43-43b0-a567-89755f11ae27.jpg: 256x256 1 x, 9.1ms
image 1182/1196 /content/drive/MyDrive/dataset-completed/test/images/x.dd4ba6e5-b60d-430e-8d43-7da761b0ccfc.jpg: 256x256 1 x, 6.8ms
image 1183/1196 /content/drive/MyDrive/dataset-completed/test/images/x.e8af08db-3b83-43f2-a367-08a9abc01bb8.jpg: 256x256 1 x, 7.3ms
image 1184/1196 /content/drive/MyDrive/dataset-completed/test/images/x.fe5f49dd-e61b-4873-93d2-14b643603151.jpg: 256x256 1 x, 7.1ms
image 1185/1196 /content/drive/MyDrive/dataset-completed/test/images/y.582b54c7-b7a8-4fdd-b04c-443059bf2b9e.jpg: 256x256 1 y, 7.5ms
image 1186/1196 /content/drive/MyDrive/dataset-completed/test/images/y.662e5687-bb8f-43fa-810a-823ca0802fbb.jpg: 256x256 1 y, 7.4ms
image 1187/1196 /content/drive/MyDrive/dataset-completed/test/images/y.e2fefe59-18c7-43fc-8435-6d1b77246e2c.jpg: 256x256 1 y, 6.7ms
image 1188/1196 /content/drive/MyDrive/dataset-completed/test/images/y.e4bcf519-53a0-4516-bc46-9e109b92f007.jpg: 256x256 1 y, 6.8ms
image 1189/1196 /content/drive/MyDrive/dataset-completed/test/images/y.ed6681e5-8fbb-483f-80db-1d7575515350.jpg: 256x256 1 y, 6.9ms
image 1190/1196 /content/drive/MyDrive/dataset-completed/test/images/y.f307320b-8915-45e7-8b23-7fd5c688420e.jpg: 256x256 1 y, 6.6ms
image 1191/1196 /content/drive/MyDrive/dataset-completed/test/images/y.f3673099-9b0c-48d1-b495-03b8fde620dc.jpg: 256x256 1 y, 9.7ms
image 1192/1196 /content/drive/MyDrive/dataset-completed/test/images/y.f47159c2-483b-4ee0-ac23-50765fd6f943.jpg: 256x256 1 y, 9.7ms
image 1193/1196 /content/drive/MyDrive/dataset-completed/test/images/y.f481b22a-a17b-44ad-acab-ebae35f9091f.jpg: 256x256 1 y, 6.7ms
image 1194/1196 /content/drive/MyDrive/dataset-completed/test/images/y.f71a9e30-0665-4c23-bee4-288aee67c5cb.jpg: 256x256 1 y, 7.4ms
image 1195/1196 /content/drive/MyDrive/dataset-completed/test/images/y.f79ff00f-2374-4fba-9932-d5b95a64b7db.jpg: 256x256 1 y, 7.5ms
image 1196/1196 /content/drive/MyDrive/dataset-completed/test/images/y.f84588ac-e289-409d-b494-23dee5482e5d.jpg: 256x256 1 y, 8.7ms
Speed: 0.2ms pre-process, 9.2ms inference, 1.3ms NMS per image at shape (1, 3, 256, 256)
Results saved to runs/detect/exp3

```

Fig. 14. Tiempo de ejecución del modelo YOLO v5 personalizado para el dataset de validación.

Speed: 0.2ms pre-process, 9.2ms inference, 1.3ms NMS per image at shape (1, 3, 256, 256)
 Total: $TT = TPP + TI + TNMS \Rightarrow 10.7ms$

Esta velocidad se detalla en la imagen 14, donde se muestra que el tiempo de procesamiento previo (pre-process) es de 0.2 milisegundos, el tiempo de inferencia (inference) es de 9.2 milisegundos y el tiempo de supresión no máxima (NMS) es de 1.3 milisegundos por imagen, considerando una forma (shape) de (1, 3, 256, 256).

El modelo logra una velocidad de detección bastante eficiente, con un tiempo total de ejecución de alrededor de 10.7 milisegundos por imagen en el conjunto de validación de 1196 imágenes. Esto indica que el modelo puede procesar rápidamente las imágenes de lenguaje de señas, lo que resulta en un rendimiento satisfactorio para su aplicación en tiempo real.

Modelo YOLO v5

```

!ls
!python detect.py --weights runs/train/yolov5s_results/weights/best.pt --img 256 --conf 0.4 --source /content/drive/MyDrive/dataset-completed/test/images
image 1141/1196 /content/drive/MyDrive/dataset-completed/test/images/u.c4604395-96b7-4747-8fde-2b398fec6aea.jpg: 256x256 1 u, 8.3ms
image 1142/1196 /content/drive/MyDrive/dataset-completed/test/images/u.c522ad12-5960-4b22-88e9-72e84cea6bda.jpg: 256x256 1 u, 10.3ms
image 1143/1196 /content/drive/MyDrive/dataset-completed/test/images/u.c8019c66-aa4b-45bf-9158-c65ca78fa279.jpg: 256x256 1 u, 9.1ms
image 1144/1196 /content/drive/MyDrive/dataset-completed/test/images/u.cb16c831-22b3-4756-aa9d-48ab8c260d66.jpg: 256x256 1 u, 9.7ms
image 1145/1196 /content/drive/MyDrive/dataset-completed/test/images/u.da95f6f1-61b7-499a-b442-9ec931e94715.jpg: 256x256 1 u, 11.7ms
image 1146/1196 /content/drive/MyDrive/dataset-completed/test/images/u.dde6070e-6d2e-4e11-8eb3-b9aa2322047e.jpg: 256x256 1 u, 10.1ms
image 1147/1196 /content/drive/MyDrive/dataset-completed/test/images/u.e3b1ae75-e424-4fab-b197-a5e3aca5630d.jpg: 256x256 1 u, 10.1ms
image 1148/1196 /content/drive/MyDrive/dataset-completed/test/images/u.fc27031-e429-4189-aeda-6704fa417797.jpg: 256x256 1 u, 10.4ms
image 1149/1196 /content/drive/MyDrive/dataset-completed/test/images/v.d72ac46d-73ad-45ee-a90d-1a6eead67ffe.jpg: 256x256 1 v, 10.2ms
image 1150/1196 /content/drive/MyDrive/dataset-completed/test/images/v.d9469b19-cfc9-4d36-96ab-3450db287efe.jpg: 256x256 1 v, 9.1ms
image 1151/1196 /content/drive/MyDrive/dataset-completed/test/images/v.d969b781-1e03-4ff2-a53d-06443e0fff55.jpg: 256x256 1 v, 10.6ms
image 1152/1196 /content/drive/MyDrive/dataset-completed/test/images/v.da474b94-fb3a-4c44-b295-588bd4424272.jpg: 256x256 1 v, 10.4ms
image 1153/1196 /content/drive/MyDrive/dataset-completed/test/images/v.da4ac18a-1743-4425-82d1-0134107a0465.jpg: 256x256 1 v, 10.2ms
image 1154/1196 /content/drive/MyDrive/dataset-completed/test/images/v.e0cd47c5-3abd-45de-8942-f5b6b204be01.jpg: 256x256 1 v, 10.0ms
image 1155/1196 /content/drive/MyDrive/dataset-completed/test/images/v.e4644c2f-4f26-4cc9-904f-f06a4055ce7c.jpg: 256x256 1 v, 10.2ms
image 1156/1196 /content/drive/MyDrive/dataset-completed/test/images/v.es2128cd-0474-4656-8c8d-79cf09f948e5.jpg: 256x256 1 v, 10.2ms
image 1157/1196 /content/drive/MyDrive/dataset-completed/test/images/v.e6bb7f1e-147b-421b-8ae8-08745380bd4f.jpg: 256x256 1 v, 10.6ms
image 1158/1196 /content/drive/MyDrive/dataset-completed/test/images/v.ee55f273-d3ca-4636-863c-81f7bdc7f848.jpg: 256x256 1 v, 10.3ms
image 1159/1196 /content/drive/MyDrive/dataset-completed/test/images/v.f8ce6277-c98c-4edc-802d-085f2ecc238f.jpg: 256x256 1 v, 10.3ms
image 1160/1196 /content/drive/MyDrive/dataset-completed/test/images/v.fafb54fc-e8d2-42fc-a36a-b3dc546bd968.jpg: 256x256 1 v, 9.8ms
image 1161/1196 /content/drive/MyDrive/dataset-completed/test/images/w.d279f81-2e2c-4397-b3e9-469dedd26e0e.jpg: 256x256 1 w, 11.4ms
image 1162/1196 /content/drive/MyDrive/dataset-completed/test/images/w.d57db29b-2256-40f9-9046-faa63a082b52.jpg: 256x256 1 w, 10.4ms
image 1163/1196 /content/drive/MyDrive/dataset-completed/test/images/w.d8de226a-573f-452c-a1b1-f232c3c44682.jpg: 256x256 1 w, 10.0ms
image 1164/1196 /content/drive/MyDrive/dataset-completed/test/images/w.d95fee7c-85d0-4707-b936-0c72b108a511.jpg: 256x256 1 w, 10.0ms
image 1165/1196 /content/drive/MyDrive/dataset-completed/test/images/w.dbbc4aac-8625-4ecc-9d59-c8db26310352.jpg: 256x256 1 w, 11.4ms
image 1166/1196 /content/drive/MyDrive/dataset-completed/test/images/w.df0c6dcb-25ac-491e-8c7f-bf28813f1051.jpg: 256x256 1 w, 9.7ms
image 1167/1196 /content/drive/MyDrive/dataset-completed/test/images/w.dff1ac58-66f1-48ba-aldc-53d954930224.jpg: 256x256 1 w, 10.2ms
image 1168/1196 /content/drive/MyDrive/dataset-completed/test/images/w.e10f5701-7aa7-45db-858c-632ef273733a.jpg: 256x256 1 w, 10.9ms
image 1169/1196 /content/drive/MyDrive/dataset-completed/test/images/w.e1cc2359-ba67-4865-89f8-2ada344a39f5.jpg: 256x256 1 w, 10.0ms
image 1170/1196 /content/drive/MyDrive/dataset-completed/test/images/w.e57d6009-02c7-4e03-b0c7-b28b6f98cb6b.jpg: 256x256 1 w, 9.4ms
image 1171/1196 /content/drive/MyDrive/dataset-completed/test/images/w.e8e4e4f2-3ea8-413a-b9ac-96d8f0d41917.jpg: 256x256 1 w, 15.5ms
image 1172/1196 /content/drive/MyDrive/dataset-completed/test/images/w.efeb9c7c-e2ff-4cab-b638-30fad4ffeaca.jpg: 256x256 1 w, 12.2ms
image 1173/1196 /content/drive/MyDrive/dataset-completed/test/images/x.142782f7-96bc-4fb5-acc6-dc900bd1a139.jpg: 256x256 1 x, 10.5ms
image 1174/1196 /content/drive/MyDrive/dataset-completed/test/images/x.267252ec-d0ed-4f3c-a1d0-901f53321d2f.jpg: 256x256 1 x, 10.8ms
image 1175/1196 /content/drive/MyDrive/dataset-completed/test/images/x.94897ba9-1906-4d50-b451-9e42854d8bdf.jpg: 256x256 1 x, 9.8ms
image 1176/1196 /content/drive/MyDrive/dataset-completed/test/images/x.ab29ea08-8965-4a84-ae0a-ab50eab76849.jpg: 256x256 1 x, 9.6ms
image 1177/1196 /content/drive/MyDrive/dataset-completed/test/images/x.b05f2637-f624-47c4-ac92-d9df3b56fff6a.jpg: 256x256 1 x, 9.2ms
image 1178/1196 /content/drive/MyDrive/dataset-completed/test/images/x.b7a10f2a-6df7-4f64-a76b-6048888d0a28.jpg: 256x256 1 x, 9.3ms
image 1179/1196 /content/drive/MyDrive/dataset-completed/test/images/x.c04b702e-82a2-408f-92b8-5e696d5f9884.jpg: 256x256 1 x, 9.6ms
image 1180/1196 /content/drive/MyDrive/dataset-completed/test/images/x.c69f46c7-85fb-44af-90ab-a82f981b37f4.jpg: 256x256 1 x, 9.2ms
image 1181/1196 /content/drive/MyDrive/dataset-completed/test/images/x.c71ed449-9d43-43b0-a567-89755f11ae27.jpg: 256x256 1 x, 12.2ms
image 1182/1196 /content/drive/MyDrive/dataset-completed/test/images/x.dd4ba6e5-b60d-430e-8d43-7da761b0ccfc.jpg: 256x256 1 x, 9.6ms
image 1183/1196 /content/drive/MyDrive/dataset-completed/test/images/x.e8af08db-3b83-43f2-a367-08a9abcdb1bb.jpg: 256x256 1 x, 9.4ms
image 1184/1196 /content/drive/MyDrive/dataset-completed/test/images/x.fe5f49dd-e61b-4873-93d2-14b643603151.jpg: 256x256 1 x, 9.4ms
image 1185/1196 /content/drive/MyDrive/dataset-completed/test/images/y.582b54c7-b7a8-4fdd-b04c-443059bf2b9e.jpg: 256x256 1 y, 9.2ms
image 1186/1196 /content/drive/MyDrive/dataset-completed/test/images/y.662e5687-bb8f-43fa-810a-823ca0902fb6.jpg: 256x256 1 y, 9.4ms
image 1187/1196 /content/drive/MyDrive/dataset-completed/test/images/y.e2feef59-18c7-43fc-8435-6d1b77246e2c.jpg: 256x256 1 y, 9.4ms
image 1188/1196 /content/drive/MyDrive/dataset-completed/test/images/y.e4bcf519-53a0-4516-bc46-9e109b92f007.jpg: 256x256 1 y, 9.1ms
image 1189/1196 /content/drive/MyDrive/dataset-completed/test/images/y.ed6681e5-8fbb-483f-80db-1d7575513350.jpg: 256x256 1 y, 9.1ms
image 1190/1196 /content/drive/MyDrive/dataset-completed/test/images/y.f30d7320-8915-45e7-8b23-7fd5c688420e.jpg: 256x256 1 y, 11.5ms
image 1191/1196 /content/drive/MyDrive/dataset-completed/test/images/y.f3673099-9b0c-48d1-b495-03b8fde620dc.jpg: 256x256 1 y, 10.2ms
image 1192/1196 /content/drive/MyDrive/dataset-completed/test/images/y.f47159c2-483b-4ee0-ac23-50765fd6f943.jpg: 256x256 1 y, 9.2ms
image 1193/1196 /content/drive/MyDrive/dataset-completed/test/images/y.f481b22a-1a7b-44ad-acab-ebae35f0091f.jpg: 256x256 1 y, 9.2ms
image 1194/1196 /content/drive/MyDrive/dataset-completed/test/images/y.f71a9e30-0665-4c23-bee4-288aee67c5cb.jpg: 256x256 1 y, 9.4ms
image 1195/1196 /content/drive/MyDrive/dataset-completed/test/images/y.f79ff00f-2374-4fba-9932-d5b95a64b7db.jpg: 256x256 1 y, 9.9ms
image 1196/1196 /content/drive/MyDrive/dataset-completed/test/images/y.f84588ac-e289-409d-409d-b494-23dee5482e5d.jpg: 256x256 1 y, 9.9ms
Speed: 0.2ms pre-process, 7.5ms inference, 1.2ms NMS per image at shape (1, 3, 256, 256)
Results saved to runs/detect/exp

```

Fig. 15. Tiempo de ejecución del modelo de YOLO v5 para el dataset de validación.

Speed: 0.2ms pre-process, 7.5ms inference, 1.2ms NMS per image at shape (1, 3, 256, 256)
 Total: TT=TPP+TI+TNMS => 8.9ms

La velocidad de detección del modelo se ha medido para un conjunto total de 1196 imágenes, donde el tiempo de procesamiento previo (pre-process) es de 0.2 milisegundos, el

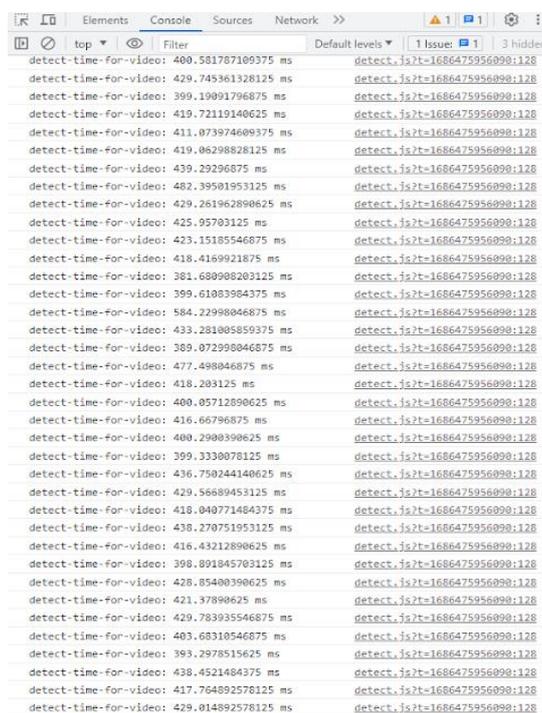
tiempo de inferencia (inference) es de 7.5 milisegundos y el tiempo de supresión no máxima (NMS) es de 1.2 milisegundos por imagen, considerando una forma (shape) de (1, 3, 256, 256).

El modelo muestra una velocidad de detección aún más rápida, con un tiempo total de ejecución de alrededor de 8.9 milisegundos por imagen en el conjunto de datos. Esto indica que el modelo puede procesar las imágenes de lenguaje de señas de manera eficiente.

Variable Dependiente

- **Tiempo de Procesamiento**

El tiempo de ejecución de la función de detección es un indicador del tiempo que tarda en procesarse cada iteración de detección en milisegundos (ms). En este caso, hemos calculado el promedio del tiempo de ejecución de la función de detección utilizando una serie de mediciones.



detect-time-for-video	detect.js?t=1686475956090:128
400.581787189375 ms	detect.js?t=1686475956090:128
429.745361328125 ms	detect.js?t=1686475956090:128
399.10991796875 ms	detect.js?t=1686475956090:128
419.72119140625 ms	detect.js?t=1686475956090:128
411.073974609375 ms	detect.js?t=1686475956090:128
419.06298828125 ms	detect.js?t=1686475956090:128
439.29296875 ms	detect.js?t=1686475956090:128
482.39581953125 ms	detect.js?t=1686475956090:128
429.261962890625 ms	detect.js?t=1686475956090:128
425.95793125 ms	detect.js?t=1686475956090:128
423.15185546875 ms	detect.js?t=1686475956090:128
418.4169921875 ms	detect.js?t=1686475956090:128
381.680988203125 ms	detect.js?t=1686475956090:128
399.61083984375 ms	detect.js?t=1686475956090:128
584.22998046875 ms	detect.js?t=1686475956090:128
433.281805859375 ms	detect.js?t=1686475956090:128
389.072998046875 ms	detect.js?t=1686475956090:128
477.498846875 ms	detect.js?t=1686475956090:128
418.203125 ms	detect.js?t=1686475956090:128
400.85712890625 ms	detect.js?t=1686475956090:128
416.66796875 ms	detect.js?t=1686475956090:128
400.2980390625 ms	detect.js?t=1686475956090:128
399.3330078125 ms	detect.js?t=1686475956090:128
436.750244140625 ms	detect.js?t=1686475956090:128
429.56589453125 ms	detect.js?t=1686475956090:128
418.040771484375 ms	detect.js?t=1686475956090:128
438.270751953125 ms	detect.js?t=1686475956090:128
416.43212890625 ms	detect.js?t=1686475956090:128
398.891845703125 ms	detect.js?t=1686475956090:128
428.85400390625 ms	detect.js?t=1686475956090:128
421.37898625 ms	detect.js?t=1686475956090:128
429.783935546875 ms	detect.js?t=1686475956090:128
403.68318546875 ms	detect.js?t=1686475956090:128
393.2978515625 ms	detect.js?t=1686475956090:128
438.4521484375 ms	detect.js?t=1686475956090:128
417.764892578125 ms	detect.js?t=1686475956090:128
429.814892578125 ms	detect.js?t=1686475956090:128

Fig. 16. Resultados del tiempo de procesamiento.

En la imagen anterior se presentan los resultados del tiempo de procesamiento, los cuales serán trasladados a un cuadro estadístico para una mejor visualización y análisis.

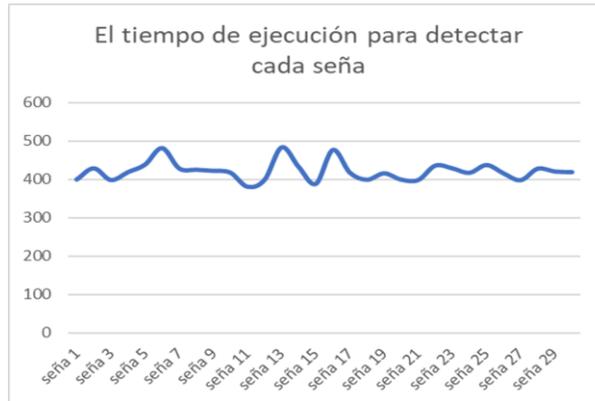


Fig. 17. Resultados del tiempo de procesamiento estadístico.

Cálculo de tiempo promedio de ejecución de detección de cada seña
Total = 17083.677 /50 => 341.67354 ms.

El cálculo del tiempo promedio de ejecución de detección para cada señal se realizó dividiendo el tiempo total de 17083.677 milisegundos entre el número total de señales, en este caso 50. Esto resulta en un tiempo promedio de ejecución de aproximadamente 341.67354 milisegundos por señal.

Este cálculo muestra el tiempo promedio que toma el modelo para detectar cada señal en el conjunto de datos, lo que proporciona una idea de la eficiencia del modelo en la detección de señales del lenguaje de señas.

- **Porcentaje de uso del CPU**

Para calcular el promedio del tiempo de CPU y la memoria utilizando el archivo “.log” proporcionado desde el navegador, podemos realizar los siguientes pasos:

Tiempo de CPU promedio:143214.42 ms

Promedio de valores de memoria:

1. **totalJSHeapSize:** (92.89) / 9 = 10.32
2. **usedJSHeapSize:** (87.45) / 9 = 9.72
3. **jsHeapSizeLimit:** (3585.82) / 9 = 398.42

El tiempo de CPU promedio de 143214.42 milisegundos revela la dedicación de la CPU al procesamiento del modelo de detección de lenguaje de señas. Respecto a la

memoria, los datos muestran que en promedio se emplean 10.32 unidades para el tamaño total del montón de JavaScript ('totalJSHeapSize'), 9.72 unidades para la memoria utilizada ('usedJSHeapSize'), y un límite máximo de 398.42 unidades para el tamaño del montón ('jsHeapSizeLimit'). Estos valores proporcionan una visión del rendimiento del modelo en términos de tiempo de CPU y uso de memoria durante la detección de señales del lenguaje de señas.

- **Tasa de errores**

Los resultados previamente presentados se utilizan en esta sección para calcular la tasa de errores del modelo seleccionado con mejor rendimiento, empleando la métrica de precisión.

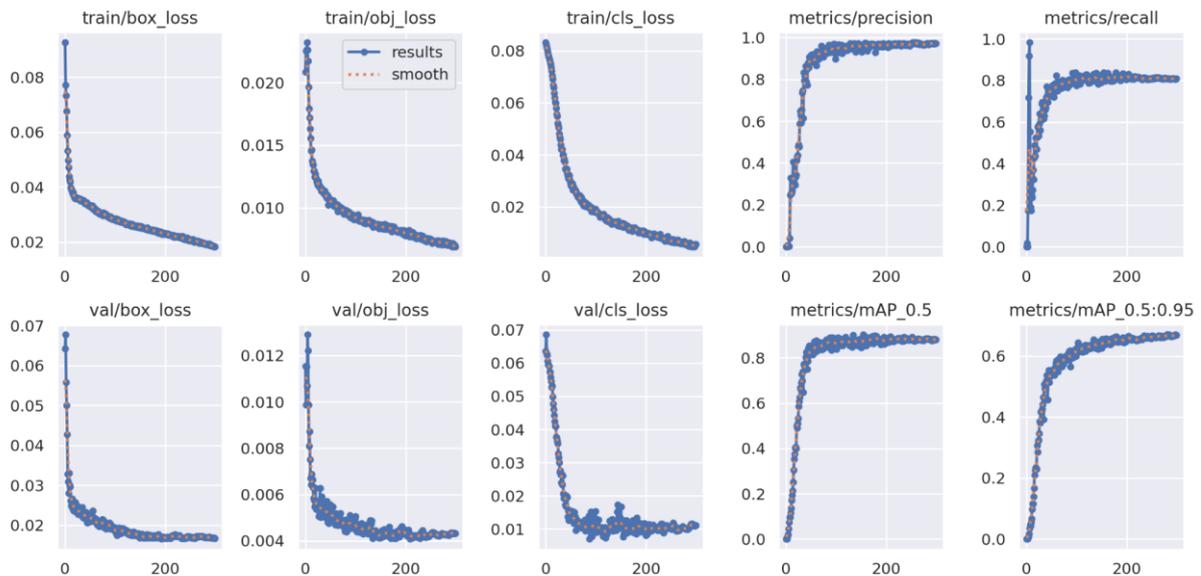


Fig. 18. Resultados del modelo basado en YOLOv5.

En la columna "metricas/precisión" se encuentra la métrica de precisión con un valor de 0.976, el cual corresponde a los valores calculados en los resultados previamente presentados.

$$\text{Tasa de errores} = 1 - 0.976 = 0.024$$

La tasa de errores, calculada restando la métrica de precisión de 1, es igual a 0.024. Esto significa que el modelo tiene una tasa de error del 2.4%, lo cual indica que el 97.6% de las señas del lenguaje de señas fueron correctamente detectadas por el modelo.

3.2. Discusión

Después de implementar los modelos y analizar los resultados, se observa que, en particular en el reconocimiento de lenguaje de señas, el modelo YOLO destaca. Investigaciones previas mencionan que YOLO tiene la capacidad de detectar objetos en imágenes a una velocidad de cuadros más rápida y con una precisión alta en comparación con otros algoritmos de detección de objetos. Basándonos en estas ventajas, se desarrolló el sistema web utilizando el modelo YOLO.

El sistema web se diseñó y construyó aprovechando las características y el rendimiento del modelo YOLO en el contexto específico del reconocimiento de alfabeto de lenguaje de señas. La implementación del sistema permitió la detección y reconocimiento de gestos y signos en tiempo real, proporcionando una interfaz amigable para los usuarios.

La elección del modelo YOLO demostró ser altamente eficiente y efectiva en el reconocimiento de lenguaje de señas, como se evidencia en los resultados obtenidos. Los valores de mAP del 90.2%, precisión de 97,6%, F1-Confidence Curva de 89,18% y recuperación de 82% reflejan un rendimiento destacado del sistema implementado.

Los resultados obtenidos en nuestra implementación del modelo YOLO muestran una alineación significativa con los resultados reportados por los autores de la investigación previa. Sin embargo, es importante destacar que nuestros resultados no logran superar los obtenidos en dicha investigación debido a la calidad de las imágenes utilizadas en nuestro proyecto.

Aunque no alcanzamos a superar los resultados de la investigación previa debido a esta limitación, la alineación de nuestros resultados con los reportados por los autores respalda la validez y confiabilidad de nuestro enfoque. Esto indica que, a pesar de las limitaciones, el modelo YOLO sigue siendo una elección sólida para el reconocimiento de lenguaje de señas.

Nuestro sistema web ha sido diseñado y desarrollado teniendo en cuenta estas limitaciones, y aun así garantiza la validez y confiabilidad de los resultados obtenidos. A pesar de las imágenes de baja calidad, hemos logrado obtener resultados prometedores que

respaldan la utilidad y efectividad de nuestro sistema en el reconocimiento de gestos y signos del lenguaje de señas.

3.3. Aporte de la investigación

En esta investigación, se enfoca en el desarrollo infantil y la educación preescolar, comenzando por la identificación de los gestos fijos que representan las letras del abecedario en el lenguaje de señas. Las señales recolectadas consisten únicamente en letras del abecedario con gestos estáticos, las cuales están detalladas en la tabla presentada en el anexo 10.

Objetivo 1: Adquirir una base de datos que contenga imágenes de letras de lenguaje de señas.

En este apartado, para la construcción de la base de datos, se utilizaron vídeos de personas realizando lenguaje de señas con el objetivo de obtener imágenes estáticas o fotogramas. Estas imágenes fueron procesadas empleando las herramientas de OpenCV y Python.

Para el procesamiento de los vídeos, se estableció que cada uno debía tener una duración de 10 a 15 segundos, además de ser grabados en diferentes entornos, condiciones de iluminación, ángulos, entre otros aspectos.

El algoritmo empleado para extraer los fotogramas de los vídeos de lenguaje de señas sigue los pasos detallados en el anexo 7. Este algoritmo fue desarrollado utilizando funciones de OpenCV en Python. Dichas funciones se encargaron de abrir los vídeos, leer cada fotograma y guardarlos como imágenes individuales en formato JPG, como se muestra en la figura 18.

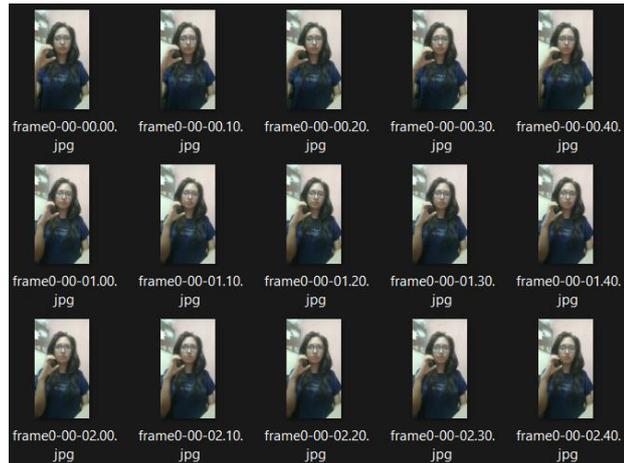


Fig. 19. Imágenes con formato .jpg adquiridas por el algoritmo.

Una vez obtenidos los fotogramas de los videos de lenguaje de señas, se procedió a reducir el tamaño de las imágenes a 256 x 256 píxeles con el objetivo de disminuir la cantidad de memoria necesaria para su procesamiento. En el artículo [56], se menciona que el tamaño de la imagen tiene un impacto directo en los recursos computacionales requeridos, así como en el tiempo de entrenamiento del modelo.

Posteriormente, a cada imagen se le asignó un identificador único, lo cual facilitó su posterior procesamiento y análisis. Esta tarea se llevó a cabo mediante el uso de herramientas de Python, como la librería OpenCV, que permiten trabajar con imágenes y añadir información adicional a ellas. De esta manera, se generó un conjunto de imágenes etiquetadas y listas para su uso en LabelImg.

Durante el proceso de etiquetado de las imágenes o fotogramas utilizando la herramienta LabelImg, se llevó a cabo de forma gráfica la creación de cajas delimitadoras. Esto implica identificar y localizar las señas en las imágenes, para luego ser etiquetadas y guardadas en archivos txt o xml correspondientes. La asignación de los archivos se realizó en función del modelo a entrenar, como se detalla en la tabla 6 siguiente.

TABLA VI. FORMATOS DE ARCHIVOS DE LOS MODELOS

Modelo	Archivo
Yolov5	txt
MobileNet V2	xml

En la información de los archivos se incluyen anotaciones como la posición y el tamaño de los objetos, las cuales son empleadas para el entrenamiento de los modelos de aprendizaje. En la figura 19, se observa que cada imagen está asociada con un archivo de texto (txt) que detalla las características necesarias para el entrenamiento del modelo.

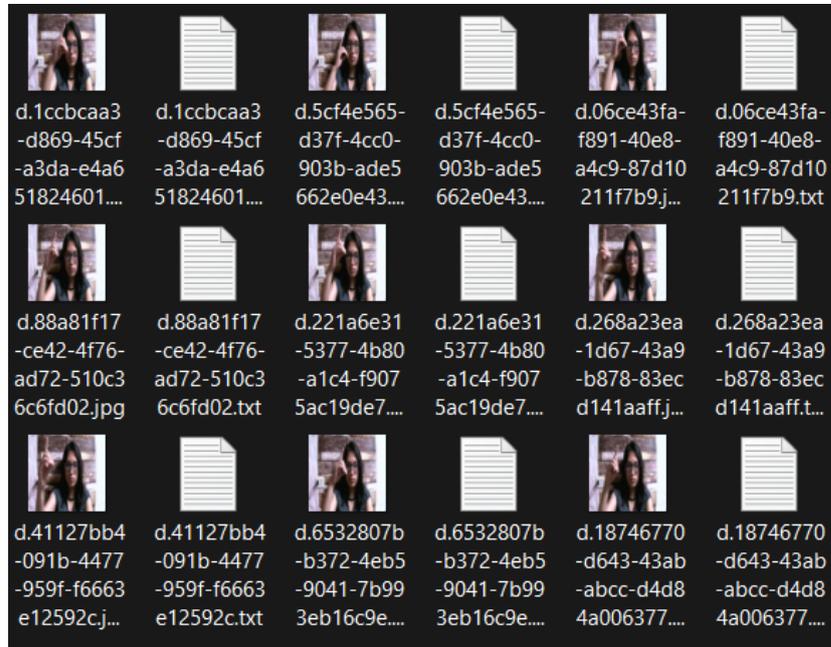


Fig. 20. Formato txt adquirido por el Labellmg.

Todos los pasos para el procesamiento de imágenes en Labellmg y la asignación de etiquetas a cada imagen están detallados en el anexo 8. Además, se han creado dos carpetas donde se encuentran las imágenes de prueba y las imágenes de entrenamiento. Posteriormente, las imágenes se dividieron en conjuntos con un porcentaje del 70% para entrenamiento, 20% para pruebas y 10% para validación.

TABLA VII. DISTRIBUCIÓN DE LAS CARPETAS PARA EL ENTRENAMIENTO.

Carpetas	Porcentaje	Contenido
Train	70%	Imágenes Labels
Test	20%	Imágenes Labels
Validation	10%	Imágenes labels

Nota: Organización de las carpetas para el entrenamiento para la implementación del modelo

En el procesamiento de las imágenes se crean las anotaciones del dataset en un archivo protobuf (.pbtxt).

```
## Ejemplo de descripciones de labels.  
item  
{  
  name: 'a'  
  id: 1  
}
```

Luego, se crearon los archivos utilizando el script generate-record.py, tal como se detalla en el anexo 12.

Objetivo 2: Establecer e implementar modelos de aprendizaje profundo para la traducción del lenguaje de señas.

Para la elección del modelo de aprendizaje profundo, se consideraron los métodos que han demostrado mejores resultados en la traducción de lenguajes de señas, según lo que se hizo una revisión de artículos que mencionan los modelos utilizados para el reconocimiento del lenguaje de señas.

A continuación, se presentará un cuadro comparativo de los modelos seleccionados, junto con sus respectivos porcentajes de precisión.

TABLA VIII. MODELOS QUE PRESENTAN MEJORES RESULTADOS.

Nº	Modelo	Dataset	Tipo de Lenguaje de seña	Precisión	Autores
[31]	ResNet18, MobileNet_V2 y EfficientNet_B1	Imágenes de letras del abecedario y números en lengua de señas	Lengua de signos boliviana (BdSL)	99.99%	Podder, et al
[57]	ResNet	Palabras en Lenguaje de Señas y Frases utilizadas en situaciones de emergencia	Lenguaje de Signos Coreano	84,5%	Choi et al
[18]	Convolutional Neural Networks (CNN)	Letras del Alfabeto en Gestos de Mano	Indian Sign Language (ISL)	99%	Vashisth et al
[26]	MobileNetV2 y ResNet18	Videos de gestos del Alfabeto y Expresiones en Lenguaje de Señas	lenguaje de signos árabe	87.69%	Podder et al.
[19]	CNN	Imágenes de letras del abecedario en lengua de señas	Lenguaje de signos americano (ASL)	99.05%	Hama Rawf et al.
[25]	YOLO	Imágenes de letras del abecedario en lengua de señas	Lenguaje de signos americano (ASL)	81.76%	Rivera-Acosta
[27]	YOLO	Imágenes de letras del abecedario y números en lengua de señas	lenguaje de señas americano	96%	Buttar et al.
[58]	Mobile Net V2	Imágenes de letras del abecedario en lengua de señas	Lenguaje de signos americano (ASL)	98,67%	Kin Yun Lum, Yeh Huann Goh, Yi Bin Lee

Tras analizar la información de los artículos, se observa que los modelos que muestran una mayor precisión en el reconocimiento de los lenguajes son ResNet18, CNN MobileNet y YOLO, en ese orden, tal como se muestra en la tabla. No obstante, existen diversos procesos que contribuyen a alcanzar estos resultados en los modelos. Por lo tanto, se consideró examinar el modelo que mejor se ajusta al desarrollo del sistema web.

Según [29], los modelos utilizados fueron CNNs pre-entrenadas como ResNet18, MobileNet_V2 y EfficientNet_B1. El entrenamiento de estos modelos se realizó tanto con imágenes de fondo como sin ellas, lo que permitió identificar que el modelo entrenado con imágenes de fondo fue más efectivo. Para [19] el modelo CNN demostró una capacidad excepcional para reconocer señales kurdas en tiempo real. Esto se debió a su

preprocesamiento, el cual incluyó el reescalado de imágenes para normalizar los valores de píxeles. En el artículo de [58] el modelo MobileNet puede ser implementado en una aplicación de smartphone para mejorar la comunicación entre personas sordas y mudas y el público en general por su procesamiento de transmisiones de cámara en vivo. En el artículo mencionado [27], el modelo YOLO señala que puede procesar vídeos a una velocidad alta, como 28.9 fps. Esto le permite capturar detalles con mayor precisión y rapidez, logrando una alta precisión en la identificación de señas estáticas en imágenes fijas.

Por consiguiente, luego de analizar los diversos modelos y compararlos en un cuadro comparativo, se optó por el modelo YOLO debido a su alta precisión y capacidad para procesar vídeos a una velocidad superior en comparación con los demás.

Modelo YOLO V5 Personalizado

Con el objetivo de implementar YOLO, se decidió utilizar PyTorch, un marco de trabajo de aprendizaje profundo muy conocido, y se diseñó un modelo personalizado de YOLOv5 en su versión pequeña. Tras estudiar la arquitectura y los parámetros, se inició la redacción del código en un archivo YAML para definir la estructura del modelo.

Debido a que no tomaremos el modelo que viene como pre definido, le haremos algunos cambios que creemos que pueden mejorar el rendimiento de este.

El primer paso consistió en establecer el número de clases, las cuales en este caso eran las diferentes letras del lenguaje de señas. Luego, se ajustaron los multiplicadores de profundidad y ancho para adaptar el modelo a las necesidades del proyecto.

Yolo v5s	Yolo v5s personalizado
<pre data-bbox="220 1697 751 1800"># Parameters nc: 80 # number of classes depth_multiple: 0.33 # model depth multiple width_multiple: 0.50 # layer channel multiple</pre>	<pre data-bbox="837 1697 1315 1800"># parameters nc: {num_classes} # number of classes depth_multiple: 0.33 # model depth multiple width_multiple: 0.50 # layer channel multiple</pre>

El modelo original utiliza una combinación de capas Conv y C3 (bloques convolucionales con conexiones residuales), mientras que el modelo personalizado

utilizaremos BottleneckCSP, que es una variante de bloques convolucionales más densos y complejos.

En ambos modelos se realiza el submuestreo en ciertos puntos del backbone, pero el número y tipo de capas utilizadas para este propósito pueden diferir, lo que afecta la resolución espacial y la profundidad de las características extraídas.

Las anclas representaban cajas de referencia importantes para predecir los gestos en las imágenes. Se definieron tres conjuntos de anclas que abarcaban diferentes escalas y niveles de resolución, con el fin de asegurar la captura de los gestos en diversos contextos.

Posteriormente, se enfocan en construir el backbone, la parte central del modelo. Se empleó una combinación de capas convolucionales, capas de foco y capas BottleneckCSP para extraer características significativas de las imágenes. Esto permitiría capturar las sutilezas y detalles de los gestos del lenguaje de señas.

El siguiente componente fue el head, encargado de tomar las características extraídas por el backbone y generar las predicciones finales. Se empleó una combinación de capas convolucionales, capas de concatenación y capas BottleneckCSP para fusionar las características de diferentes niveles de resolución y realizar predicciones precisas de los gestos.

TABLA IX. HIPER PARÁMETROS DE YOLOV5.

Nombre de hiper parámetro	Descripción	Valor
lr0	Tasas de aprendizaje inicial	0.01
lrf	Tasas de aprendizaje final	0.01
momentum	Influencia de gradientes anteriores en la actualización de pesos.	0.937
weight_decay	Término de regularización para evitar el sobreajuste.	0.0005
warmup_epochs	Ajustes iniciales de momentum.	3.0
warmup_momentum	Ajustes tasa de aprendizaje.	0.8

warmup_bias_lr	Ajustes el sesgo.	0.1
box		0.05
cls	Pesos para los diferentes componentes de la función de pérdida.	0.5
cls_pw		1.0
obj_pw		1.0
iou_t	Umbral de superposición IOU mínimo	0.2
anchor_t	selección de anclas	4.0
fl_gamma	Ajuste del factor focal para el desequilibrio de clases	0.0
hsv_h		0.015
hsv_s	Aleatoriedad en la modificación del espacio de color HSV.	0.7
hsv_v		0.4
degrees		0.0
translate		0.1
scale	Aumentación geométrica aplicada a las imágenes.	scale=0.5
shear		0.0
perspective		0.0
flipud		0.0
fliplr		0.5
mosaic		1.0
mixup	Probabilidades de aplicar técnicas de aumentación de datos	0.0
copy_paste		0.0

Nota: configuraciones para entrenar el modelo YOLOV5

Para entrenar el modelo, se tuvo en cuenta el tamaño de las imágenes de entrada, el tamaño del lote y el número de épocas. Las imágenes se redimensionaron a un tamaño de

416 x 416 píxeles y se decidió trabajar con un tamaño de lote de 16 imágenes. Para garantizar un entrenamiento adecuado del modelo, se establecieron 300 épocas.

Además, se utilizó un archivo YAML de datos que contenía la configuración de los datos de entrenamiento, incluyendo la ruta de las imágenes y las clases de los gestos del lenguaje de señas.

Finalmente, se aseguró de que el modelo se inicializa correctamente. Dado que se estaba construyendo el modelo desde cero, no se utilizaron pesos pre-entrenados.

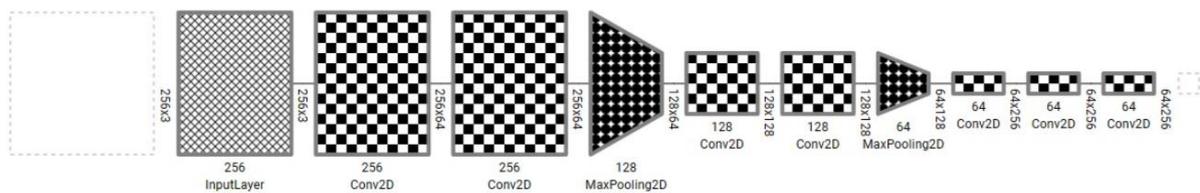


Fig. 21. MODELO YOLOV5

MobileNet V2

Para el entrenamiento, se aplicó el modelo MobileNetV2 para construir una red profunda y ligera. En MobileNetV2, la capa central factoriza la convolución en profundidad y la convolución en puntos. En la convolución en profundidad, se aplica un filtro a cada canal de entrada, lo que reduce la cantidad de parámetros necesarios para entrenar la red neuronal. Este proceso se puede observar en la figura 21.

Para la implementación de MobileNet, primero se configuró los parámetros a continuación se muestra lo señalado:

1. Se estableció el número de clases en el modelo según la cantidad de etiquetas (clases) que se tienen en el conjunto de datos.
2. Se definió el tamaño del lote (batch size) utilizado durante el entrenamiento.
3. Se especificó la ubicación del punto de control pre-entrenado utilizado para la transferencia de aprendizaje.
4. Se configuró el tipo de punto de control pre-entrenado como "detection".
5. Se especificó la ruta del archivo de mapa de etiquetas (label map).

6. Se configuró la ruta del archivo de registros TFRecord de entrenamiento.
7. Se estableció la ruta del archivo de mapa de etiquetas (label map) para la evaluación del modelo.
8. Se especificó la ruta del archivo de registros TFRecord de prueba (evaluación).
9. Se guardó la configuración en un archivo de configuración en formato de texto.

Se definieron los parámetros, los cuales se ubicarán en las características del extractor

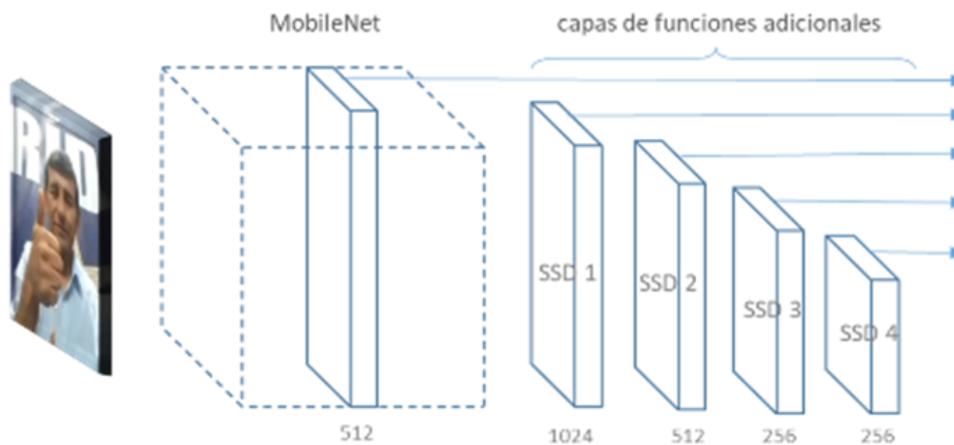


Fig. 22. Implementación del modelo MobileNet V2.

TABLA X. HIPER PARÁMETROS DE MOBILE NET V2.

Nombre de hiper parámetro	Descripción	Valor
type	Se utiliza el modelo MobileNet V2 con FPN (Feature Pyramid Network) implementado en Keras	"mobilenet_v2_fpn_keras"
depth_multiplier	Se establece un multiplicador de profundidad de 1.0, lo que significa que la profundidad de la red se mantiene igual	1.0
min_depth	Se establece una profundidad mínima de 16 para asegurar que haya suficientes canales en las capas convolucionales	16
conv_hyperparams.regularizer.l2_regularizer.weight	Se utiliza la regularización L2 con un peso de $3.9999998989515007e-05$ para penalizar los valores de los pesos excesivamente grandes.	3.9999998989515007e-05
conv_hyperparams.initializer.random_normal_initializer.mean	Se utiliza una inicialización de pesos con distribución normal, con un valor de media de 0.0 y una desviación con valor estándar de 0.009999999776482582.	0.0

conv_hyperparams.initializer.random_normal_initializer.stddev	RELU_6: Se aplica la función de activación	0.00999999 977648258 2
conv_hyperparams.activation	Se aplica la normalización por lotes (batch normalization) en las capas convolucionales, con un factor de decaimiento de 0.996999979019165, permitiendo que sea de mejor manera la adaptación a los datos.	RELU_6
conv_hyperparams.batch_norm.decay		0.99699997 9019165
conv_hyperparams.batch_norm.scale		true
conv_hyperparams.batch_norm.epsilon		0.00100000 004749745 13
use_depthwise	Se utiliza la convolución depthwise para reducir el costo computacional y el número de parámetros.	true
override_base_feature_extractor_hyperparams	Se sobrescriben los hiper parámetros del extractor de características base, lo que permite ajustarlos específicamente para este modelo.	true
fpn.min_level	Se establece el nivel mínimo de la pirámide de características en 3	3
fpn.max_level	Se establece el nivel máximo de la pirámide de características en 7	7
fpn.additional_layer_depth	Se agrega una capa adicional en la pirámide de características con una profundidad de 128	128

Nota: configuraciones para entrenar el modelo MobileNetV2

Se decidió realizar 5000 épocas para permitir que el modelo aprendiera, como se detalla en el anexo 13.

Objetivo 3: 3.5.3 Desarrollar un sistema web de videollamadas con la integración del modelo con mejor rendimiento implementado

En este apartado, cuyo objetivo es desarrollar el sistema web que implementó el modelo con mejor rendimiento, se utilizarán algunas librerías necesarias para leer el modelo exportado, en este caso YOLOv5, mediante el uso de React.js. Estas librerías pueden ser instaladas con los comandos adecuados.

```
npm install @tensorflow/tfjs
```

En TensorFlow.js, la limpieza de memoria es esencial para evitar la acumulación de recursos y la sobrecarga en el navegador. Durante el procesamiento de datos en el aprendizaje automático, se utilizan tensores, que son estructuras de datos fundamentales para representar y manipular datos multidimensionales. Cada vez que se crean nuevos tensores en TensorFlow.js, se asigna memoria para almacenar los datos correspondientes. Sin una gestión adecuada de la memoria, esto puede llevar a una utilización excesiva de recursos y, eventualmente, a bloqueos del navegador.

Por lo tanto, es importante liberar explícitamente la memoria una vez que los tensores ya no sean necesarios. En TensorFlow.js, se proporciona el método `dispose()` para liberar la memoria de los tensores. Al llamar a `dispose()` en un tensor, se liberan los recursos asociados y se hace espacio para la creación de nuevos tensores. Esto ayuda a mantener la aplicación eficiente y evita la acumulación de recursos innecesarios.

```
tf.ready().then(async () => {  
  
  const yolov5 = await tf.loadGraphModel(  
    `${window.location.href}/${modelName}_web_model/model.json`,  
    {  
      onProgress: (fractions) => {  
        setLoading({ loading: true, progress: fractions });  
      },  
    }  
  );  
  
  const dummyInput = tf.ones(yolov5.inputs[0].shape);  
  const warmupResult = await yolov5.executeAsync(dummyInput);  
  
  tf.dispose(warmupResult);  
  
  tf.dispose(dummyInput);  
  
  setLoading({ loading: false, progress: 1 });  
  
  setModel({  
    net: yolov5,
```

```
inputShape: yolov5.inputs[0].shape,  
});  
});
```

Uso de WebGL

TensorFlow.js utiliza WebGL como backend para el cálculo numérico y las operaciones matemáticas intensivas en el aprendizaje automático.

Se utiliza WebGL para proporcionar un rendimiento mucho más rápido en comparación con el procesamiento basado en CPU estándar. WebGL aprovecha la capacidad que se tiene para el procesamiento en paralelo de las unidades de procesamiento de gráficos (GPU) de la tarjeta gráfica del usuario, lo que permite realizar cálculos matemáticos complejos de manera eficiente. Esto es especialmente importante para las aplicaciones de aprendizaje automático que involucran operaciones intensivas, como la ejecución de modelos de redes neuronales.

Al utilizar WebGL como backend en TensorFlow.js, se puede aprovechar la aceleración de GPU para obtener un rendimiento optimizado y tiempos de procesamiento más rápidos en comparación con la ejecución en CPU.

```
npm install @tensorflow/tfjs-backend-webgl  
import "@tensorflow/tfjs-backend-webgl";
```

Para poder llegar a usar WebGL en el navegador es necesario activar todas las opciones desde las propiedades del navegador.

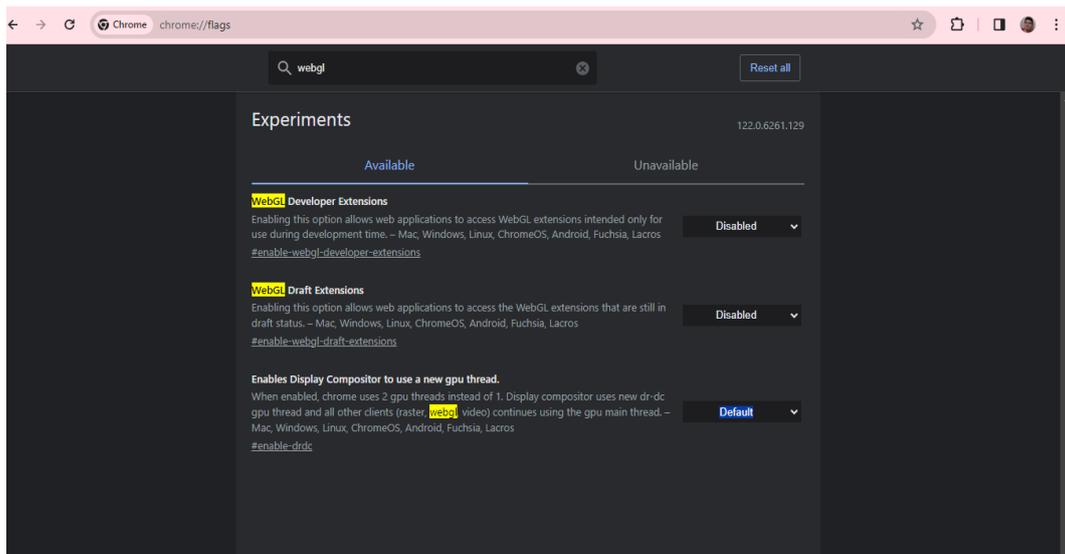


Fig. 23. Configuraciones para WebGL en el navegador.

Archivos binarios de modelos:

En TensorFlow.js, los modelos se guardan en archivos JSON y se acompañan de archivos binarios que contienen los pesos y los parámetros del modelo. El archivo JSON proporciona la estructura del modelo en un formato legible por máquina, mientras que los archivos binarios contienen los datos numéricos necesarios para realizar inferencias.

```
python export.py --weights yolov5/runs/train/yolov5s_results*/weights/best.pt --include saved_model tfjs
```


Objetivo 4: Analizar los resultados de las técnicas implementadas de lenguaje de señas.

En el desarrollo de la aplicación web para analizar los resultados del modelo de lenguaje de señas, se implementaron una serie de funcionalidades clave utilizando React.js. Una de las áreas en las que se centraron fue el rendimiento de la aplicación. Para medir el rendimiento, se utilizó la API del navegador llamada 'performance'. Esta API proporciona información valiosa sobre el tiempo de CPU y el uso de memoria.

```
setInterval(() => {  
  
  console.log("Medición de CPU y memoria:");  
  
  const cpuTime = performance.now();  
  
  console.log("Tiempo de CPU utilizado:", cpuTime.toFixed(2), "ms");  
  
  const memoryInfo = {  
  
    totalJSHeapSize: (console.memory.totalJSHeapSize / 1048576).toFixed(2), // Convertir a MB  
    usedJSHeapSize: (console.memory.usedJSHeapSize / 1048576).toFixed(2), // Convertir a MB  
    jsHeapSizeLimit: (console.memory.jsHeapSizeLimit / 1048576).toFixed(2), // Convertir a MB  
  };  
  
  console.log("Información de memoria:");  
  
  console.table(memoryInfo);  
  
}, interval)
```

Para obtener datos específicos sobre la memoria, utilizamos la consola y algunas propiedades proporcionadas por 'console.memory'. La información que recopilamos incluye el tamaño total de la memoria asignada para el montón de JavaScript ('totalJSHeapSize'), la cantidad de memoria en uso en el montón de JavaScript ('usedJSHeapSize') y el límite máximo de memoria asignado para el montón de JavaScript ('jsHeapSizeLimit'). Estos datos

nos permitieron tener una mejor comprensión del consumo de memoria de nuestra aplicación y nos ayudaron a optimizar su rendimiento.

En cuanto a la detección del modelo de TensorFlow en la aplicación web, también se utilizaron algunas técnicas para medir el tiempo de ejecución y evitar problemas de memoria. Se implementó una función llamada 'detectVideo' que se encarga de procesar cada fotograma del video utilizando TensorFlow.js. Para medir el tiempo de detección del modelo, se utilizó la función 'console.time()' al inicio del proceso y 'console.timeEnd()' al final.

```
const detectFrame = async () => {  
  console.time("detect-time-for-video");  
  
  // Código de limpieza de canvas  
  //.....  
  
  setTimeout(detectFrame,1000);  
  
  tf.engine().endScope();  
  
  console.timeEnd("detect-time-for-video");  
  
};
```

Dentro de la función 'detectVideo', se verifica si la video fuente tiene un ancho válido. En caso contrario, se limpia el lienzo y se detiene el proceso para evitar errores. Luego, se realiza el preprocesamiento del fotograma para adaptarlo al tamaño requerido por el modelo de TensorFlow. A continuación, se ejecuta el modelo utilizando el método 'executeAsync' y se obtienen las cajas delimitadoras, las puntuaciones de confianza y las clases de los objetos detectados.

Después de renderizar las cajas delimitadoras en el lienzo, es importante liberar la memoria para evitar la acumulación innecesaria de recursos. Se utiliza 'tf.dispose()' para eliminar los tensores generados durante la ejecución del modelo.

Además, se utilizó 'setTimeout()' para obtener el siguiente fotograma del video y procesarlo de manera recursiva. Esto permitió mantener una detección continua de los

objetos en el video. Finalmente, se empleó 'tf.engine().endScope()' para finalizar el alcance de la ejecución de TensorFlow y liberar los recursos asociados.

Con el fin de tener un mayor rendimiento en la videollamada, es necesario activar la aceleración por hardware del navegador, que está se encuentra en configuración.

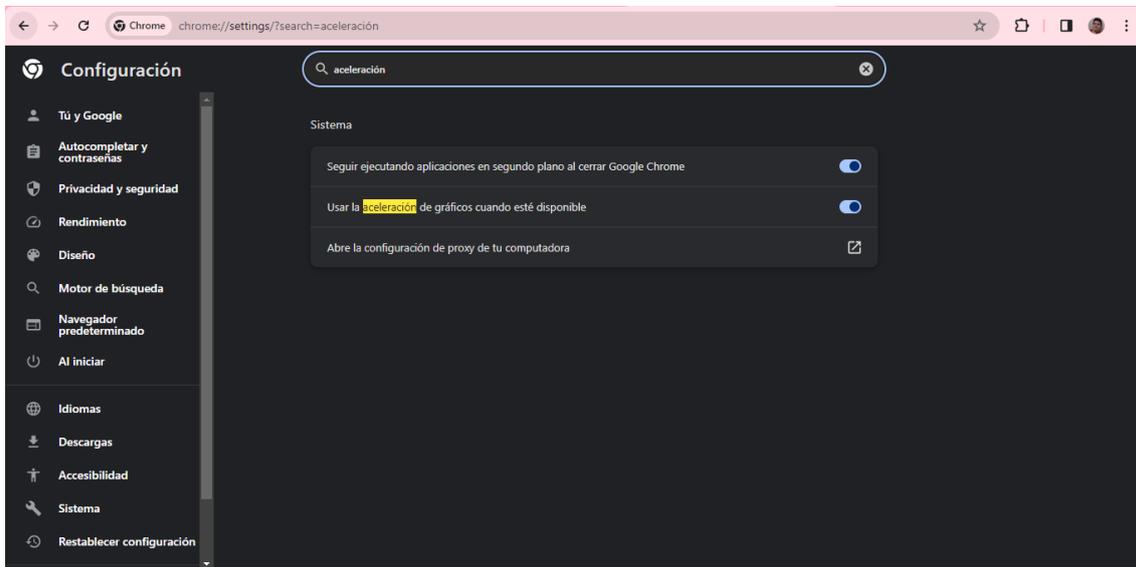


Fig. 26. Ubicación de aceleración por hardware.

A continuación, se presenta la aplicación web donde se podrá practicar y aprender el lenguaje de señas peruano.

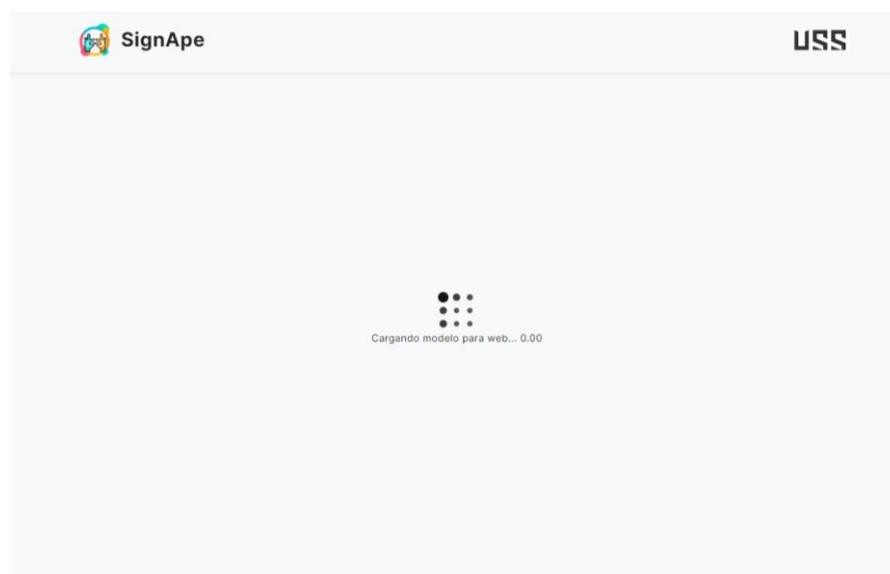


Fig. 27. Carga de modelo en la aplicación web con diferentes usuarios

Tu ID de usuario es:

weathered-violet-4806

O Copia el ID de tu amigo aquí:

Nombre de usuario



Nicolette Isis Pacheco Contreras y
Andy Josue Santisteban Ostos

Fig. 28. Carga de modelo en la aplicación web con diferentes usuarios



Fig. 29. Aplicación web "" - transmisión de videos en diferentes aplicaciones.

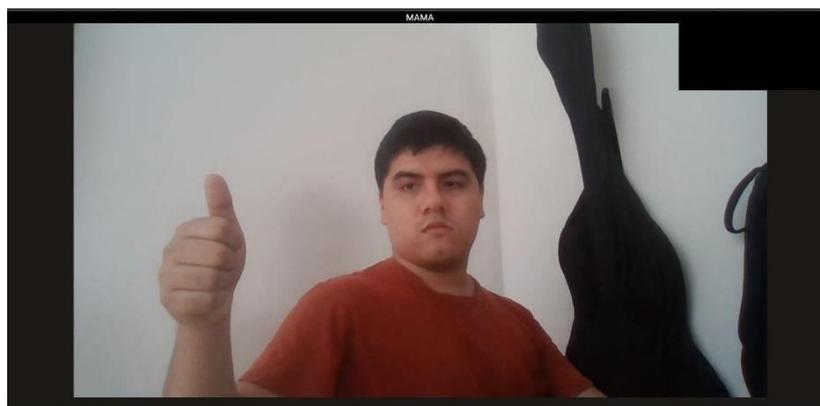


Fig. 30. Aplicación web "<https://signape.onrender.com/>" - Formando palabras con letras.

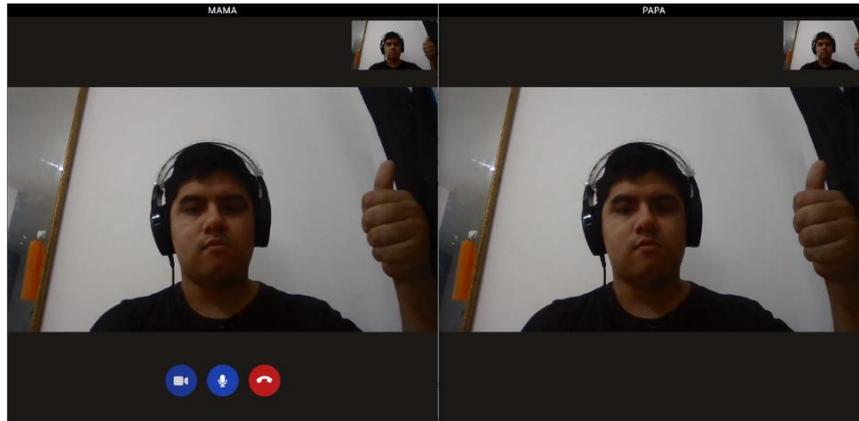


Fig. 31. Aplicación web "<https://signape.onrender.com/>" - Formando palabras con letras

IV. CONCLUSIONES Y RECOMENDACIONES

4.1. Conclusiones

Se adquirió una base de datos constituida por 3,596 imágenes del abecedario del Lenguaje de Señas Peruano (LSP). Esta base de datos incluyó diferentes condiciones de iluminación, contraste y cambios en el fondo, lo que hizo que el modelo fuera más robusto ante estas características.

Se realizó un estudio bibliográfico profundo de los modelos de redes neuronales empleados para la identificación del lenguaje de señas. Se encontró que los dos más utilizados que se ajustan al desarrollo del sistema web son MobileNet y YOLO. El primero es una arquitectura ligera que puede ser implementada dentro de una aplicación móvil, mientras que el segundo se destaca por permitir una mayor mAP, especialmente con la versión v5, la cual ofrece un modelo pre-entrenado con imágenes para identificar manos en diferentes posiciones.

El modelo fue modificado agregando una última capa convolucional adicional y entrenado con las imágenes adquiridas para identificar los 27 signos de letras y 10 de dígitos obteniendo el valor medio de precisión media 90.2%.

La evaluación del sistema web, determinó que era necesario tomar capturas de la webcam en un tiempo real para que el reconocimiento de la señal fuera factible. Estas capturas luego serían procesadas utilizando el modelo YOLO, lo cual proporciona una

detección coherente de la señal. Se estimó que el tiempo promedio de ejecución para cada seña, desde la captura hasta la detección, es de 341 milisegundos.

La integración de React.js y Express.js en el desarrollo de la aplicación web ha permitido la creación de una interfaz modular y escalable, facilitando la integración del modelo de detección del LSP. Esta combinación de tecnologías brinda una experiencia de usuario fluida y una arquitectura de software robusta.

La utilización de WebGL ha sido fundamental para aprovechar la capacidad de procesamiento acelerado por hardware, lo que ha permitido una detección eficiente y rápida del LSP en tiempo real. Esto ha contribuido a una experiencia visual enriquecedora y una comunicación más efectiva en las videollamadas.

La adaptabilidad de la aplicación web para su uso en una aplicación móvil basada en web ha ampliado su alcance y accesibilidad, brindando a los usuarios la posibilidad de acceder a la detección del LSP desde sus dispositivos móviles. Esto ha mejorado la portabilidad y la disponibilidad del sistema en diversos entornos.

El enfoque de desarrollo de software basado en React.js y Express.js ha demostrado ser altamente escalable y flexible, lo que ha facilitado la incorporación de nuevas funcionalidades y la adaptación del sistema a diferentes requisitos del LSP. Esto garantiza una mayor versatilidad y una respuesta eficiente a las necesidades de los usuarios.

El modelo YOLOv5 fue integrado a la aplicación web, empleando Tensorflow.js proporcionado un marco sólido para la detección del LSP en tiempo real y permitiendo una interacción más natural y fluida en las videollamadas.

Si bien se han logrado resultados destacados en la detección del LSP en tiempo real, es importante tener en cuenta las limitaciones inherentes del hardware del usuario y la calidad de la conexión de red, ya que pueden afectar el rendimiento y la precisión de la detección en ciertos escenarios. Estos aspectos deben considerarse al utilizar la aplicación en diferentes contextos y entornos.

La implementación exitosa de este sistema basado en un modelo de red neuronal convolucional YOLOv5, utilizando tecnologías de desarrollo web modernas, demuestra el

potencial de la inteligencia artificial, en particular del aprendizaje automático, en la mejora de la comunicación y la accesibilidad del Lenguaje de Señas Peruano en aplicaciones de videollamadas y transmisión de video.

Este sistema ofrece una solución innovadora y efectiva para la detección del LSP en tiempo real en un entorno web, lo que brinda oportunidades para mejorar la comunicación y la inclusión de las personas que utilizan este lenguaje.

4.2. Recomendaciones

Realizar investigaciones adicionales para evaluar el impacto de diferentes arquitecturas de red neuronal convolucional en la detección de letras y palabras del Lenguaje de Señas Peruano (LSP). Se recomienda comparar YOLOv5 con otras arquitecturas basadas en YOLO y explorar otras librerías como single-shot detector (SSD) para determinar cuál ofrece el mejor rendimiento en este contexto específico.

Realizar estudios de rendimiento exhaustivos para evaluar la escalabilidad del sistema implementado en términos de la cantidad de usuarios concurrentes y la carga de procesamiento. Se recomienda utilizar métricas como el tiempo de respuesta y la utilización de recursos del servidor para identificar posibles cuellos de botella y optimizar el rendimiento general del sistema.

Realizar pruebas exhaustivas de usabilidad y evaluaciones de experiencia de usuario para evaluar la eficacia y la satisfacción de los usuarios con el sistema implementado. Se recomienda utilizar métodos de evaluación estándar, como cuestionarios de satisfacción del usuario y pruebas de usabilidad, para recopilar datos cuantitativos y cualitativos que permitan identificar áreas de mejora y optimización.

Investigar y desarrollar técnicas de mejora de la precisión del modelo de detección del LSP mediante el uso de técnicas de aumento de datos específicas para el LSP, como la variación de ángulos de mano y gestos, así como la incorporación de datos sintéticos generados por computadora para aumentar la diversidad del conjunto de datos de entrenamiento.

Explorar la viabilidad de utilizar técnicas de aprendizaje activo para mejorar el modelo

de detección del LSP en un entorno web, lo cual implica la interacción con expertos del LSP para etiquetar y proporcionar retroalimentación sobre las detecciones incorrectas, lo que permitirá un entrenamiento continuo y adaptativo del modelo.

Investigar la posibilidad de utilizar técnicas de interpretación y explicabilidad de modelos de inteligencia artificial para comprender y visualizar mejor las decisiones de detección del LSP. Esto permitirá a los usuarios y desarrolladores comprender las razones detrás de las detecciones y promover la confianza en el sistema implementado.

Realizar estudios comparativos de las tecnologías de desarrollo web utilizadas en el proyecto, como React.js y Express.js, para evaluar su efectividad y eficiencia en términos de rendimiento, escalabilidad y facilidad de desarrollo. Se recomienda comparar estas tecnologías con alternativas disponibles para garantizar la elección óptima en futuras implementaciones.

Investigar y desarrollar estrategias para garantizar la seguridad y la privacidad de los usuarios en el entorno web y las videollamadas. Esto implica la implementación de técnicas de cifrado de extremo a extremo y la adopción de prácticas de seguridad recomendadas para proteger los datos sensibles de los usuarios durante la transmisión y el procesamiento.

REFERENCIAS

- [1] World Health Organization, «Más de 360 millones de personas padecen pérdida de audición discapacitante, según las nuevas estimaciones mundiales publicadas por la Organización Mundial de la Salud.», WHO. Accedido: 23 de abril de 2023. [En línea]. Disponible en: <https://www.who.int/es/news/item/27-02-2013-millions-have-hearing-loss-that-can-be-improved-or-prevented>
- [2] WASLI, «World Federation for the Deaf (WFD) - WASLI», WASLI - World Association of Sign Language Interpreters. Accedido: 23 de abril de 2023. [En línea]. Disponible en: <https://wasli.org/special-interest/world-federation-for-the-deaf-wfd>
- [3] World Health Organization, «Sordera y pérdida de la audición». Accedido: 23 de abril de 2023. [En línea]. Disponible en: <https://www.who.int/es/news-room/fact-sheets/detail/deafness-and-hearing-loss>
- [4] A. B. Domínguez, «Educación para la inclusión de alumnos sordos», *Rev. Latinoam. Educ. Inclusiva*, p. 17, 2017.
- [5] C. McClain-Nhlapo, «To achieve 'learning for all', we must create inclusive systems for students with disabilities». Accedido: 26 de abril de 2023. [En línea]. Disponible en: <https://blogs.worldbank.org/education/achieve-learning-all-we-must-create-inclusive-systems-students-disabilities>
- [6] I. A. Zrigat y Y. M. Al-Smadi, «Challenges of Educating Students who are Deaf and Hard-of-Hearing in Jordan», abr. 2012, [En línea]. Disponible en: https://www.researchgate.net/publication/236866061_Challenges_of_Educating_Students_who_are_Deaf_and_Hard-of-Hearing_in_Jordan
- [7] Volver a escuchar, «Consecuencias psicológicas de la pérdida auditiva - Volver a escuchar». Accedido: 22 de abril de 2023. [En línea]. Disponible en: <https://volvieraescuchar.com/consecuencias-psicologicas-de-la-perdida-auditiva/>
- [8] Human Rights Watch, «El lenguaje de señas, un componente clave para los derechos de las personas sordas», Human Rights Watch. Accedido: 23 de abril de 2023. [En línea]. Disponible en: <https://www.hrw.org/es/news/2018/09/23/el-lenguaje-de-senas-un-componente-clave-para-los-derechos-de-las-personas-sordas>
- [9] M. Á. Martínez Sánchez, A. Muñoz-García, y C. Ros Gil, «Perception of the Impact of COVID-19 on a Sample of Spaniards with Hearing Disabilities», *Int. J. Environ. Res. Public. Health*, vol. 20, n.º 2, p. 1460, ene. 2023, doi: 10.3390/ijerph20021460.
- [10] Visualfy, «Estudiantes sordos y clases online: ¿más barreras durante el Covid-19?», Visualfy. Accedido: 23 de abril de 2023. [En línea]. Disponible en: <https://www.visualfy.com/es/estudiantes-sordos-y-educacion-online-mas-barreras-durante-el-covid19/>
- [11] Defensoría del Pueblo, «Defensoría del Pueblo: debe facilitarse el aprendizaje de la lengua de señas peruana y promover la identidad lingüística y cultural de las personas sordas», Defensoría del Pueblo - Perú. Accedido: 22 de abril de 2023. [En línea]. Disponible en: <https://www.defensoria.gob.pe/defensoria-del-pueblo-debe-facilitarse-el-aprendizaje-de-la-lengua-de-senas-peruana-y-promover-la-identidad-linguistica-y-cultural-de-las-personas-sordas/>
- [12] S. La República, «COVID en Perú: ¿qué mascarillas utilizan las personas con discapacidad auditiva? | Sociedad | La República». Accedido: 23 de abril de 2023. [En línea]. Disponible en: <https://larepublica.pe/sociedad/2022/02/02/covid-en-peru-que-mascarillas-utilizan-las-personas-con-discapacidad-auditiva>
- [13] A. Rodríguez, «Sordera en el Perú, ¿qué se está haciendo al respecto?», Copera Infancia. Accedido: 23 de abril de 2023. [En línea]. Disponible en: <https://coperainfancia Peru.com/2021/09/22/sordera-en-el-peru-que-se-esta-haciendo-al-respecto/>
- [14] J. L. Bermudez, «Signslator: traductor en línea de español a lenguaje de signos», Soft & Apps. Accedido: 30 de abril de 2023. [En línea]. Disponible en: <https://www.softandapps.info/2014/05/29/signslator-traductor-lenguaje-de-signos/>

- [15] Martagamez, «Showleap, el primer traductor de lenguaje de signos», Hablemos de empresas. Accedido: 30 de abril de 2023. [En línea]. Disponible en: <https://hablemosdeempresas.com/pymes/showleap/>
- [16] J. R. Trillo, F. Trillo, y F. Jiménez, «Software de aprendizaje de las letras del lenguaje de signos mediante leap motion», p. 9, 2021.
- [17] C. Lu, M. Kozakai, y L. Jing, «Sign Language Recognition with Multimodal Sensors and Deep Learning Methods», *Electronics*, vol. 12, n.º 23, p. 4827, nov. 2023, doi: 10.3390/electronics12234827.
- [18] H. K. Vashisth, T. Tarafder, R. Aziz, M. Arora, y A. Alpana, «Hand Gesture Recognition in Indian Sign Language Using Deep Learning», en *RAiSE-2023*, MDPI, dic. 2023, p. 96. doi: 10.3390/engproc2023059096.
- [19] K. M. H. Hama Rawf, A. O. Abdulrahman, y A. A. Mohammed, «Improved Recognition of Kurdish Sign Language Using Modified CNN», *Computers*, vol. 13, n.º 2, p. 37, ene. 2024, doi: 10.3390/computers13020037.
- [20] M. Al-Hammadi *et al.*, «Deep Learning-Based Approach for Sign Language Gesture Recognition With Efficient Hand Gesture Representation», *IEEE Access*, vol. 8, pp. 192527-192542, 2020, doi: 10.1109/ACCESS.2020.3032140.
- [21] R. Rastgoo, K. Kiani, y S. Escalera, «Sign Language Recognition: A Deep Survey», *Expert Syst. Appl.*, vol. 164, p. 113794, feb. 2021, doi: 10.1016/j.eswa.2020.113794.
- [22] N. C. Camgoz, S. Hadfield, O. Koller, H. Ney, y R. Bowden, «Neural Sign Language Translation», en *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT: IEEE, jun. 2018, pp. 7784-7793. doi: 10.1109/CVPR.2018.00812.
- [23] T. Ananthanarayana *et al.*, «Deep Learning Methods for Sign Language Translation», *ACM Trans. Access. Comput.*, vol. 14, n.º 4, pp. 1-30, dic. 2021, doi: 10.1145/3477498.
- [24] S.-K. Ko, C. J. Kim, H. Jung, y C. Cho, «Neural Sign Language Translation Based on Human Keypoint Estimation», *Appl. Sci.*, vol. 9, n.º 13, p. 2683, jul. 2019, doi: 10.3390/app9132683.
- [25] M. Rivera-Acosta, J. M. Ruiz-Varela, S. Ortega-Cisneros, J. Rivera, R. Parra-Michel, y P. Mejia-Alvarez, «Spelling Correction Real-Time American Sign Language Alphabet Translation System Based on YOLO Network and LSTM», *Electronics*, vol. 10, n.º 9, p. 1035, abr. 2021, doi: 10.3390/electronics10091035.
- [26] K. K. Podder *et al.*, «Signer-Independent Arabic Sign Language Recognition System Using Deep Learning Model», *Sensors*, vol. 23, n.º 16, p. 7156, ago. 2023, doi: 10.3390/s23167156.
- [27] A. M. Buttar, U. Ahmad, A. H. Gumaei, A. Assiri, M. A. Akbar, y B. F. Alkhamees, «Deep Learning in Sign Language Recognition: A Hybrid Approach for the Recognition of Static and Dynamic Signs», *Mathematics*, vol. 11, n.º 17, p. 3729, ago. 2023, doi: 10.3390/math11173729.
- [28] I. Papastratis, C. Chatzikonstantinou, D. Konstantinidis, K. Dimitropoulos, y P. Daras, «Artificial Intelligence Technologies for Sign Language», *Sensors*, vol. 21, n.º 17, p. 5843, ago. 2021, doi: 10.3390/s21175843.
- [29] M. C. Madahana, K. Khoza-Shangase, N. Moroe, D. Mayombo, O. Nyandoro, y J. Ekoru, «A proposed artificial intelligence-based real-time speech-to-text to sign language translator for South African official languages for the COVID-19 era and beyond: In pursuit of solutions for the hearing impaired», *S. Afr. J. Commun. Disord.*, vol. 69, n.º 2, ago. 2022, doi: 10.4102/sajcd.v69i2.915.
- [30] I. A. Adeyanju, O. O. Bello, y M. A. Adegboye, «Machine learning methods for sign language recognition: A critical review and analysis», *Intell. Syst. Appl.*, vol. 12, p. 200056, nov. 2021, doi: 10.1016/j.iswa.2021.200056.
- [31] K. K. Podder *et al.*, «Bangla Sign Language (BdSL) Alphabets and Numerals Classification Using a Deep Learning Model», *Sensors*, vol. 22, n.º 2, p. 574, ene. 2022, doi: 10.3390/s22020574.
- [32] P. Dönmez, «Introduction to Machine Learning, 2nd ed., by Ethem Alpaydın. Cambridge, MA: The MIT Press 2010. ISBN: 978-0-262-01243-0. \$54/£ 39.95 + 584 pages.», *Nat.*

- Lang. Eng.*, vol. 19, n.º 2, pp. 285-288, abr. 2013, doi: 10.1017/S1351324912000290.
- [33] T. Hastie, R. Tibshirani, y J. Friedman, *The Elements of Statistical Learning*. en Springer Series in Statistics. New York, NY: Springer New York, 2009. doi: 10.1007/978-0-387-84858-7.
- [34] W. Ertel, *Introduction to Artificial Intelligence*. en Undergraduate Topics in Computer Science. Cham: Springer International Publishing, 2017. doi: 10.1007/978-3-319-58487-4.
- [35] E. Alpaydin, *Introduction to machine learning*, Third edition. en Adaptive computation and machine learning. Cambridge, Massachusetts: The MIT Press, 2014.
- [36] Na, «¿Cómo funcionan las Convolutional Neural Networks? Visión por Ordenador», *Aprende Machine Learning*. Accedido: 7 de mayo de 2023. [En línea]. Disponible en: <https://www.aprendemachinelarning.com/como-funcionan-las-convolutional-neural-networks-vision-por-ordenador/>
- [37] A. Mosavi, S. Ardabili, y A. R. Várkonyi-Kóczy, «List of Deep Learning Models», en *Engineering for Sustainable Future*, vol. 101, A. R. Várkonyi-Kóczy, Ed., en Lecture Notes in Networks and Systems, vol. 101. , Cham: Springer International Publishing, 2020, pp. 202-214. doi: 10.1007/978-3-030-36841-8_20.
- [38] I. Goodfellow, Y. Bengio, y A. Courville, *Deep learning*. en Adaptive computation and machine learning. Cambridge, Massachusetts: The MIT Press, 2016.
- [39] M. Abadi *et al.*, «TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems». arXiv, 16 de marzo de 2016. Accedido: 3 de mayo de 2023. [En línea]. Disponible en: <http://arxiv.org/abs/1603.04467>
- [40] G. Van Rossum, *Python Tutorial*. 1995.
- [41] «LabelImg». [En línea]. Disponible en: <https://github.com/heartexlabs/labelimg>
- [42] D. Smilkov *et al.*, «TensorFlow.js: Machine Learning for the Web and Beyond», p. 13.
- [43] F. Chollet, *Deep learning with Python*. Shelter Island, New York: Manning Publications Co, 2018.
- [44] TensorFlow, «What's coming in TensorFlow 2.0», TensorFlow. Accedido: 7 de mayo de 2023. [En línea]. Disponible en: <https://medium.com/tensorflow/whats-coming-in-tensorflow-2-0-d3663832e9b8>
- [45] «WebGL», The Khronos Group. Accedido: 11 de junio de 2023. [En línea]. Disponible en: <https://www.khronos.org/>
- [46] J. Redmon, S. Divvala, R. Girshick, y A. Farhadi, «You Only Look Once: Unified, Real-Time Object Detection». arXiv, 9 de mayo de 2016. doi: 10.48550/arXiv.1506.02640.
- [47] datahacker.rs, «#OD1 YOLO Object Detection», Master Data Science. Accedido: 5 de mayo de 2023. [En línea]. Disponible en: <https://datahacker.rs/how-to-peform-yolo-object-detection-using-keras/>
- [48] «WebSockets - Referencia de la API Web | MDN». Accedido: 7 de mayo de 2023. [En línea]. Disponible en: https://developer.mozilla.org/es/docs/Web/API/WebSockets_API
- [49] «P2P and the W3C». [En línea]. Disponible en: <https://www.w3.org/2001/04/w3c-p2p/>
- [50] R. Quintanar-Casillas y Ma. S. Hernández-López, «Modelos Tecnológicos de Aprendizaje Adaptativo Aplicados a la Educación», *Rev. Tecnológica-Educ. Docentes* 20, vol. 15, n.º 1, pp. 41-66, jun. 2022, doi: 10.37843/rted.v15i1.308.
- [51] V. Mehmood, «What is Web RTC meaning and How does it work? Explained!», PUREVPN. Accedido: 3 de mayo de 2023. [En línea]. Disponible en: <https://www.purevpn.com/internet-privacy/what-is-webrtc>
- [52] «About | Node.js», Node.js. Accedido: 3 de mayo de 2023. [En línea]. Disponible en: <https://nodejs.org/en/about>
- [53] A. Mardan, *Pro Express.js*. [New York, NY] New York, NY: Apress Springer Science+Business Media, 2014.
- [54] S. Aggarwal, «Modern Web-Development using ReactJS», vol. 5, n.º 1, p. 5, 2018.
- [55] H. Sampieri, C. F. Collado, y P. B. Lucio, «METODOLOGÍA DE LA INVESTIGACIÓN», p. 4, 2008.
- [56] M. Tan y Q. V. Le, «EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks». arXiv, 11 de septiembre de 2020. doi: 10.48550/arXiv.1905.11946.

- [57] S.-G. Choi, Y. Park, y C.-B. Sohn, «Dataset Transformation System for Sign Language Recognition Based on Image Classification Network», *Appl. Sci.*, vol. 12, n.º 19, p. 10075, oct. 2022, doi: 10.3390/app121910075.
- [58] K. Y. Lum, Y. H. Goh, y Y. B. Lee, «American Sign Language Recognition Based on MobileNetV2», *Adv. Sci. Technol. Eng. Syst. J.*, vol. 5, n.º 6, pp. 481-488, nov. 2020, doi: 10.25046/aj050657.
- [59] S. Xie, R. Girshick, P. Dollár, Z. Tu, y K. He, «Aggregated Residual Transformations for Deep Neural Networks», en *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI: IEEE, jul. 2017, pp. 5987-5995. doi: 10.1109/CVPR.2017.634.
- [60] C. Szegedy *et al.*, «Going deeper with convolutions», en *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, MA, USA: IEEE, jun. 2015, pp. 1-9. doi: 10.1109/CVPR.2015.7298594.
- [61] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, y Z. Wojna, «Rethinking the Inception Architecture for Computer Vision». arXiv, 11 de diciembre de 2015. doi: 10.48550/arXiv.1512.00567.
- [62] S.-H. Tsang, «Review: Xception — With Depthwise Separable Convolution, Better Than Inception-v3 (Image...», Medium. Accedido: 7 de mayo de 2023. [En línea]. Disponible en: <https://towardsdatascience.com/review-xception-with-depthwise-separable-convolution-better-than-inception-v3-image-dc967dd42568>
- [63] S. Zagoruyko y N. Komodakis, «Wide Residual Networks», 2016, doi: 10.48550/ARXIV.1605.07146.
- [64] M. Tan *et al.*, «MnasNet: Platform-Aware Neural Architecture Search for Mobile», en *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA, USA: IEEE, jun. 2019, pp. 2815-2823. doi: 10.1109/CVPR.2019.00293.
- [65] N. Ma, X. Zhang, H.-T. Zheng, y J. Sun, «ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design», en *Computer Vision – ECCV 2018*, vol. 11218, V. Ferrari, M. Hebert, C. Sminchisescu, y Y. Weiss, Eds., Cham: Springer International Publishing, 2018, pp. 122-138. doi: 10.1007/978-3-030-01264-9_8.
- [66] A. Krizhevsky, I. Sutskever, y G. E. Hinton, «ImageNet classification with deep convolutional neural networks», *Commun. ACM*, vol. 60, n.º 6, pp. 84-90, may 2017, doi: 10.1145/3065386.
- [67] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, y K. Keutzer, «SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size». arXiv, 4 de noviembre de 2016. doi: 10.48550/arXiv.1602.07360.
- [68] K. Simonyan y A. Zisserman, «Very Deep Convolutional Networks for Large-Scale Image Recognition». arXiv, 10 de abril de 2015. doi: 10.48550/arXiv.1409.1556.
- [69] G. Huang, Z. Liu, L. Van Der Maaten, y K. Q. Weinberger, «Densely Connected Convolutional Networks», en *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI: IEEE, jul. 2017, pp. 2261-2269. doi: 10.1109/CVPR.2017.243.
- [70] K. He, X. Zhang, S. Ren, y J. Sun, «Deep Residual Learning for Image Recognition», en *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA: IEEE, jun. 2016, pp. 770-778. doi: 10.1109/CVPR.2016.90.
- [71] J. Hu, L. Shen, y G. Sun, «Squeeze-and-Excitation Networks», en *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT: IEEE, jun. 2018, pp. 7132-7141. doi: 10.1109/CVPR.2018.00745.
- [72] B. Zoph, V. Vasudevan, J. Shlens, y Q. V. Le, «Learning Transferable Architectures for Scalable Image Recognition», en *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT: IEEE, jun. 2018, pp. 8697-8710. doi: 10.1109/CVPR.2018.00907.
- [73] S. Sabour, N. Frosst, y G. E. Hinton, «Dynamic Routing Between Capsules». arXiv, 7 de noviembre de 2017. Accedido: 7 de mayo de 2023. [En línea]. Disponible en: <http://arxiv.org/abs/1710.09829>
- [74] D. Yoo, S. Park, J.-Y. Lee, A. S. Paek, y I. S. Kweon, «AttentionNet: Aggregating Weak

- Directions for Accurate Object Detection», en *2015 IEEE International Conference on Computer Vision (ICCV)*, Santiago, Chile: IEEE, dic. 2015, pp. 2659-2667. doi: 10.1109/ICCV.2015.305.
- [75] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, y S. Belongie, «Feature Pyramid Networks for Object Detection», en *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI: IEEE, jul. 2017, pp. 936-944. doi: 10.1109/CVPR.2017.106.
- [76] Y. Lecun, L. Bottou, Y. Bengio, y P. Haffner, «Gradient-based learning applied to document recognition», *Proc. IEEE*, vol. 86, n.º 11, pp. 2278-2324, nov. 1998, doi: 10.1109/5.726791.
- [77] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, y L.-C. Chen, «MobileNetV2: Inverted Residuals and Linear Bottlenecks», en *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT: IEEE, jun. 2018, pp. 4510-4520. doi: 10.1109/CVPR.2018.00474.
- [78] K. He, G. Gkioxari, P. Dollár, y R. Girshick, «Mask R-CNN», en *2017 IEEE International Conference on Computer Vision (ICCV)*, Venice: IEEE, oct. 2017, pp. 2980-2988. doi: 10.1109/ICCV.2017.322.
- [79] T.-Y. Lin, P. Goyal, R. Girshick, K. He, y P. Dollár, «Focal Loss for Dense Object Detection», *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, n.º 2, pp. 318-327, feb. 2020, doi: 10.1109/TPAMI.2018.2858826.
- [80] J. Redmon y A. Farhadi, «YOLO9000: Better, Faster, Stronger», en *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI: IEEE, jul. 2017, pp. 6517-6525. doi: 10.1109/CVPR.2017.690.
- [81] J. Redmon y A. Farhadi, «YOLOv3: An Incremental Improvement». arXiv, 8 de abril de 2018. Accedido: 7 de mayo de 2023. [En línea]. Disponible en: <http://arxiv.org/abs/1804.02767>
- [82] S. Ren, K. He, R. Girshick, y J. Sun, «Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks», *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, n.º 6, pp. 1137-1149, jun. 2017, doi: 10.1109/TPAMI.2016.2577031.
- [83] W. Liu *et al.*, «SSD: Single Shot MultiBox Detector», en *Computer Vision – ECCV 2016*, vol. 9905, B. Leibe, J. Matas, N. Sebe, y M. Welling, Eds., en *Lecture Notes in Computer Science*, vol. 9905. , Cham: Springer International Publishing, 2016, pp. 21-37. doi: 10.1007/978-3-319-46448-0_2.
- [84] J. Long, E. Shelhamer, y T. Darrell, «Fully convolutional networks for semantic segmentation», en *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, MA, USA: IEEE, jun. 2015, pp. 3431-3440. doi: 10.1109/CVPR.2015.7298965.
- [85] O. Ronneberger, P. Fischer, y T. Brox, «U-Net: Convolutional Networks for Biomedical Image Segmentation», en *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, vol. 9351, N. Navab, J. Hornegger, W. M. Wells, y A. F. Frangi, Eds., Cham: Springer International Publishing, 2015, pp. 234-241. doi: 10.1007/978-3-319-24574-4_28.
- [86] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, y A. L. Yuille, «DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs», *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, n.º 4, pp. 834-848, abr. 2018, doi: 10.1109/TPAMI.2017.2699184.
- [87] L.-C. Chen, A. Hermans, G. Papandreou, F. Schroff, P. Wang, y H. Adam, «MaskLab: Instance Segmentation by Refining Object Detection with Semantic and Direction Features», en *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT: IEEE, jun. 2018, pp. 4013-4022. doi: 10.1109/CVPR.2018.00422.

ANEXOS



ANEXO 01: DECLARACIÓN JURADA DE ORIGINALIDAD

Quienes suscriben la DECLARACIÓN JURADA, somos **Egresados** del Programa de Estudios de **Ingeniería de Sistemas** de la Universidad Señor de Sipán S.A.C, declaramos bajo juramento que somos autores del trabajo titulado:

IMPLEMENTACIÓN DE UN MODELO DE APRENDIZAJE PROFUNDO PARA LA TRADUCCIÓN DE LENGUAJES DE SEÑAS

El texto de mi trabajo de investigación responde y respeta lo indicado en el Código de Ética del Comité Institucional de Ética en Investigación de la Universidad Señor de Sipán, conforme a los principios y lineamientos detallados en dicho documento, en relación con las citas y referencias bibliográficas, respetando el derecho de propiedad intelectual, por lo cual informo que la investigación cumple con ser inédito, original y autentico.

En virtud de lo antes mencionado, firman:

Santisteban Ostos Andy Josue	DNI: 71985693	
Pacheco Contreras Nicolette Isis	DNI: 77416740	

Pimentel, 02 de Julio de 2023.

ANEXO 02: ACTA DE REVISIÓN DE SIMILITUD DE LA INVESTIGACIÓN

Yo **Mag. Mejia Cabrera Heber Ivan** docente del curso de **Investigación II** del Programa de Estudios de **Ingeniería de Sistemas** y revisor de la investigación los estudiantes, **Santisteban Ostos Andy Josue, Pacheco Contreras Nicolette Isis**, titulada:

**IMPLEMENTACIÓN DE UN MODELO DE APRENDIZAJE PROFUNDO PARA LA
TRADUCCIÓN DE LENGUAJES DE SEÑAS**

Se deja constancia que la investigación antes indicada tiene un índice de similitud del **14%**, verificable en el reporte final del análisis de originalidad mediante el software de similitud TURNITIN. Por lo que se concluye que cada una de las coincidencias detectadas no constituyen plagio y cumple con lo establecido en la Directiva sobre índice de similitud de los productos académicos y de investigación en la Universidad Señor de Sipán S.A.C., aprobada mediante Resolución de Directorio N° 145-2022/PD-USS.

En virtud de lo antes mencionado, firma:

Mejia Cabrera Heber Ivan	DNI: 41639565	
--------------------------	---------------	---

Pimentel, 19 de Julio de 2023.

ANEXO 03: ACTA APROBACIÓN DEL ASESOR

Yo **Forero Vargas Manuel Guillermo**, quien suscribe como asesor designado mediante Resolución de Facultad N° **0758-2022/FIAU-USS**, del proyecto de investigación titulado **Implementación de un modelo de aprendizaje profundo para la traducción de lenguajes de señas**, desarrollado por los estudiantes: **Santisteban Ostos Andy Josue**, **Pacheco Contreras Nicolette Isis**., del programa de estudios de **Ingeniería de sistemas**, acredito haber revisado, realizado observaciones y recomendaciones pertinentes, encontrándose expedito para su revisión por parte del docente del curso.

En virtud de lo antes mencionado, firman:

Forero Vargas Manuel Guillermo	DNI: AV702661	
Santisteban Ostos Andy Josue	DNI: 71985693	
Pacheco Contreras Nicolette Isis	DNI: 77416740	

Pimentel, 02 de julio de 2023



ACTA DE REVISIÓN DE ASESORÍA

Yo **Forero Vargas Manuel Guillermo**, quien suscribe como asesor designado mediante Resolución de Facultad N° ° **0758-2022/FIAU-USS**, del proyecto de investigación titulado **IMPLEMENTACIÓN DE UN MODELO DE APRENDIZAJE PROFUNDO PARA LA TRADUCCIÓN DE LENGUAJES DE SEÑAS.**, desarrollado por los estudiantes: **Santisteban Ostos Andy Josue y Pacheco Contreras Nicolette Isis**, del programa de estudios de **Ingeniería de Sistemas**, acredito haber revisado, realizado observaciones y recomendaciones pertinentes tal como se detalla en el siguiente cuadro:

Fecha de revisión:	Modalidad de Asesoría:	Medio de Asesoría:	Veredicto de Asesoría:
30 Junio 2023	Virtual	https://applications.zoom.us/jti/rich/home/recording/detail	Aprobado

En virtud de lo antes mencionado, firman:

Forero Vargas Manuel Guillermo	DNI: AV702661	
Santisteban Ostos Andy Josue	DNI: 71985693	
Pacheco Contreras Nicolette Isis	DNI: 77416740	

Pimentel, 02 de julio de 2023

FICHA DE REVISIÓN

N°	Sección del Informe observado <i>(Seleccione una opción)</i>	Número de página observado	Comentario de la observación	Estado de la Observación <i>(Seleccione una opción)</i>
1	Todo el documento	4-56	Revisar redacción, verificando coincidencia de tiempos verbales, errores de tipografía y fluidez del texto.	Corregido
2	1.5.1 Lenguaje de señas	9	Pasar el listado de lenguajes de señas a un anexo. En su lugar, mencionar que se conocen más de (número) diferentes lenguajes de señas, siendo uno de ellos el peruano.	Corregido
3	1.4 Objetivos	8	Revisar la redacción del objetivo general. Reducir los objetivos 2 y 3 a uno solo.	Corregido
4	Conclusiones	57 a 61	Reescribir conclusiones observando que aparezcan como consecuencia de los objetivos propuestos y soportados por los resultados obtenidos en el trabajo.	Corregido
5	Programa	-	Aumentar la base de datos de manera que incluya más imágenes en donde las condiciones de iluminación y contraste hagan más difícil la detección de la mano, para conseguir una mayor robustez de la técnica y una mayor exactitud.	Corregido

ANEXO 04: MODELO DE RED NEURONAL

TABLA XI. MODELOS DE RED NEURONAL DE APRENDIZAJE PROFUNDO.

N	Modelo de red neuronal convolucional	Autor	Descripción
1	Resnext101_32x8d	Propuesto en el artículo "Aggregated Residual Transformation for Deep Neural Networks"	Es una variante de la arquitectura ResNeXt, que tiene 101 capas y una cardinalidad de 32x8d, es efectiva en la clasificación de imágenes [59]
2	Resnext50_32x4d	Propuesto en el artículo "Aggregated Residual Transformation for Deep Neural Networks"	Es una variante de la arquitectura ResNeXt, que tiene 50 capas y una cardinalidad de 32x4d, es efectiva en la clasificación de imágenes [59]
3	Googlenet / InceptionV1	Propuesto en el artículo "Going Deeper with Convolutions"	Red neuronal profunda de 22 capas que utiliza múltiples módulos Inception para mejorar la eficiencia y la precisión de la clasificación y detección de objetos [60]
4	InceptionV2	Desarrollado por un equipo de Google y presentado en un artículo de investigación titulado "Rethinking the Inception Architecture for Computer Vision"	La versión mejorada de Inception-v2 logra un 21,2% de error en el top-1 y un 5,6% en el top-5 para un solo cultivo en la evaluación sobre la clasificación [61]
3	Inception_V3	Desarrollado por un equipo de Google y presentado en un artículo de investigación titulado "Rethinking the Inception Architecture for Computer Vision"	Arquitectura de red neuronal convolucional utiliza módulos Inception que combinan diferentes tamaños de filtros para capturar características a diferentes escalas demostrando la eficiencia en escenarios de big data y visión móvil [61]
4	Xception	Propuesto por François Chollet, el creador de Keras, y se desarrolló en Google	Red neuronal convolucional profunda con 71 capas de profundidades una versión extrema de la arquitectura Inception que a logrado la clasificación de imágenes y ha obtenido buenos resultados en competiciones de aprendizaje automático [62]
5	Wide_resnet101_2	Propuesto en "Wide Residual Networks".	Se destaca por su eficiencia computacional y su capacidad para lograr un alto rendimiento en tareas de clasificación de imágenes con un número relativamente bajo de parámetros [63]
6	Mnasnet1_0	Propuesto en estudios de investigación, como	Arquitectura que está diseñada para dispositivos móviles, se caracteriza por

		"MnasNet: Platform-Aware Neural Architecture Search for Mobile"	tener una estructura diversa de capas, lo que permite mejorar la precisión y la eficiencia del modelo [64].
7	Wide_resnet50_2	Propuesto en "Wide Residual Networks".	Es una arquitectura de red neuronal convolucional que tiene 50 capas y utiliza bloques residuales anchos que contienen varias capas convolucionales y de activación. [63]
8	ShuffleNet_v2_x0_5	Propuesto en el artículo "ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design"	La arquitectura combina convoluciones separables en profundidad y convoluciones regulares para reducir la cantidad de operaciones de multiplicación y sumas en punto flotante (FLOP) necesarias para la inferencia. [65]
9	ShuffleNet	Propuesto en el artículo "ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design"	Mezclar las características de diferentes canales permitiendo que la información fluya de manera más eficiente a través de la red y mejore la precisión de la clasificación. [65]
10	Alexnet	Propuesto en el artículo "ImageNet Classification with Deep Convolutional Neural Networks"	La arquitectura consta de ocho capas, cinco de las cuales son capas convolucionales y tres son capas completamente conectadas, permite entrenar de manera más rápida y mejora la precisión de la red.[66]
11	SqueezeNet	Propuesto en el artículo "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size"	Reducir el tamaño del modelo mientras mantiene una precisión similar a la de otras arquitecturas populares como AlexNet, su principal estrategia es reemplazar filtros, disminuir el número de canales de entrada a los filtros y realizar downsampling tardío en la red para que las capas de convolución tengan mapas de encendido grandes [67]
12	Vgg19_bn	Propuesto en el artículo "Very Deep Convolutional Networks for Large-Scale Image Recognition"	Normalmente se utiliza para la clasificación de imágenes, como la clasificación de objetos en el conjunto de datos ImageNet. La adición de "_bn" significa que la red utiliza normalización de lotes (batch normalization) para mejorar la estabilidad y la velocidad de entrenamiento.[68]
13	DenseNet	Propuesto en el artículo "Densely Connected Convolutional Networks"	Reduce la cantidad de parámetros necesarios y la mejora de la propagación de características además la prevención

			del problema del gradiente desvaneciente.[69]
14	Densenet201	Propuesto en el artículo "Densely Connected Convolutional Networks"	Versión más profunda y compleja de la arquitectura DenseNet original, con 201 capas en total [69]
15	VGG11	Propuesto en el artículo "Very Deep Convolutional Networks for Large-Scale Image Recognition"	La red también incluye tres capas completamente conectadas al final para la clasificación. Aunque VGG11 es una versión más pequeña de la red VGG16 [68].
16	Vgg16_bn	Propuesto en el artículo "Very Deep Convolutional Networks for Large-Scale Image Recognition"	Ayuda a reducir la covarianza de las activaciones de la red, lo que puede mejorar la generalización y la velocidad de entrenamiento [68]
17	Resnet101	Propuesto en el artículo "Deep Residual Learning for Image Recognition"	Tiene una complejidad menor que las redes VGG-16/19 y es más preciso que el ResNet de 34 capas por un margen considerable.[70]
18	Resnet50	Propuesto en el artículo "Deep Residual Learning for Image Recognition"	El ResNet50 tiene un error de validación de los cinco primeros en un solo modelo del 6,71 %, lo que supera todos los resultados anteriores del conjunto.[71]
19	ResNet152	Propuesto en el artículo "Deep Residual Learning for Image Recognition"	El ResNet152 tiene un error de validación de los cinco primeros en un solo modelo del 5,71 %, lo que supera todos los resultados anteriores del conjunto.[70]
20	NASNet	Propuesto por un equipo de investigadores de Google en un artículo de investigación titulado "Learning Transferable Architectures for Scalable Image Recognition" en 2018	Utiliza un espacio de búsqueda de arquitectura específica para generar arquitecturas de CNN que son escalables y pueden manejar imágenes de diferentes tamaños y resoluciones [72]
21	Densenet169	Propuesto en el artículo "Densely Connected Convolutional Networks"	Conexiones directas entre todas las capas, permitiendo que cada capa reciba información de todas las capas anteriores y transmita información a todas las capas posteriores. [69]
22	VGG19	Propuesto en el artículo "Very Deep Convolutional Networks for Large-Scale Image Recognition"	Normalmente se utiliza para tareas de clasificación de imágenes y filtra de tamaño pequeño para una representación más detallada de las características de la imagen. [68]

23	SENet	Propuesto por Jie Hu, Li Shen y Gang Sun en un artículo de investigación titulado "Squeeze-and-Excitation Networks" en 2018	Arquitectura de red neuronal profunda que utiliza bloques SE (Squeeze-and-Excitation) para mejorar la capacidad de representación de la red [71]
24	Densenet121	Propuesto en el artículo "Densely Connected Convolutional Networks"	Versión más pequeña y menos compleja que DenseNet201 y DenseNet169, con 121 capas en total [69]
25	Vgg16	Propuesto en el artículo "Very Deep Convolutional Networks for Large-Scale Image Recognition"	Red neuronal convolucional profunda que se utiliza normalmente para tareas de clasificación de imágenes y utiliza una arquitectura simple y fácil de entender [68].
26	Mnasnet1_3	Propuesto en estudios de investigación, como "MnasNet: Platform-Aware Neural Architecture Search for Mobile"	Arquitectura de red neuronal convolucional diseñada para dispositivos móviles que es una versión mejorada de MnasNet1_0 y permite mejorar aún más la precisión del modelo [64]
27	CapsNet	Propuesto en un artículo de investigación titulado "Dynamic Routing Between Capsules" en 2017	Utilizar cápsulas en lugar de neuronas convencionales para representar características visuales [73]
28	AttentionNet	Propuesto en un artículo titulado AttentionNet: Aggregating Weak Directions for Accurate Object Detection	Utiliza una red neuronal convolucional para detectar objetos en imágenes y proporciona direcciones débiles para recortar la imagen y ajustar una caja delimitadora al objeto [74]
29	FPN	Propuesto en un artículo de investigación titulado "Feature Pyramid Networks for Object Detection" en 2017	Construye una pirámide de características que contenga información de diferentes escalas y resoluciones, lo que permite detectar objetos de diferentes tamaños y formas. [75]
30	ResNet	Propuesto en el artículo "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size"	Utiliza conexiones de salto permitiendo que la información fluya directamente desde una capa a otra más adelante en la red, saltando varias capas intermedias. [67]
31	LeNet-5	fue propuesto en el artículo "Gradient-Based Learning Applied to Document Recognition"	Se compone de dos capas convolucionales seguidas de dos capas completamente conectadas las cuales utilizan para el reconocimiento de caracteres[76].

32	MobileNetV2	Propuesto en el artículo "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications"	MobileNetV2 utiliza bloques de construcción Inverted Residuals que consisten en una capa de convolución lineal seguida de una capa de convolución no lineal (ReLU) y otra capa de convolución lineal que reduce la dimensionalidad de los datos en las capas intermedias de la red para reducir la carga computacional.[77]
33	MobileNetV3	Propuesto en el artículo "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications"	Utiliza bloques de construcción llamados "bottlenecks" que tienen una estructura similar a los "Inverted Residuals" de MobileNetV2, pero con algunas mejoras para mejorar la eficiencia y la precisión [77]
34	MobileNet	Propuesto en el artículo "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications"	Divide la convolución en dos etapas separadas para reducir el costo computacional además utiliza varios marcos de trabajo de aprendizaje profundo, como TensorFlow [77]
35	EfficientNet	Propuesto en el artículo "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks"	Se adapta a diferentes tamaños, al ancho y la resolución de la red para lograr un mejor equilibrio entre la precisión y la eficiencia computacional y conjuntos de datos [56]
36	EfficientNetB0-B7	Propuesto en el artículo "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks"	Modelo base y se puede escalar a modelos más grandes y precisos, como EfficientNet-B7, mientras se mantiene la eficiencia de los recursos.[56]
37	YOLOv1	Propuesto en el artículo "You Only Look Once: Unified, Real-Time Object Detection"	Detecta objetos en tiempo real que utilizan una única red neuronal convolucional para predecir las cajas delimitadoras y las clases de los objetos en una imagen [46]
38	DenseNet	Propuesta en el artículo "Densely Connected Convolutional Networks"	Combina las características de todas las capas anteriores en una sola capa logrado resultados de vanguardia en varios conjuntos de datos de clasificación de imágenes [69]
39	Mask R-CNN	Propuesto por Kaiming He, Georgia Gkioxari, Piotr Dollár y Ross Girshick en un artículo	Modelo de segmentación de instancias de última generación que extiende el modelo de detección de objetos Faster R-CNN [78]

		de investigación titulado "Mask R-CNN" en 2017	
40	RetinaNet	Propuesto en un artículo de investigación titulado "Focal Loss for Dense Object Detection" en 2018	Detector de objetos de una sola etapa que utiliza una red neuronal convolucional completamente convolucional (FCN) con una estructura de pirámide de características (FPN) y una subred de clasificación y regresión de cajas. [79]
41	YOLOv2	Propuesto en un artículo de investigación titulado "YOLO9000: Better, Faster, Stronger" en 2017	Detector de objetos en tiempo real capaz de operar a diferentes resoluciones de imagen, lo que permite un equilibrio entre velocidad y precisión [80].
42	YOLOv3	Propuesto en un artículo de investigación titulado "YOLOv3: An Incremental Improvement" en 2018	Enfoque de detección de objetos basado en la predicción de cajas delimitadoras en tres escalas diferentes.[81]
43	Faster R-CNN	Propuesto en un artículo de investigación titulado "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks"	Sistema de detección de objetos que consta de dos módulos: una red neuronal convolucional completamente profunda que propone regiones y un detector que utiliza las regiones propuestas [82]
44	SSD	Propuesto en un artículo de investigación titulado "SSD: Single Shot MultiBox Detector" en 2016	Utiliza la técnica MultiBox para generar las cajas delimitadoras y las probabilidades de clase de los objetos en una imagen que han sido ampliamente utilizados en aplicaciones de visión por computadora, como la detección de objetos en imágenes y videos [83].
45	FCN	Propuesto en un artículo de investigación titulado "Fully Convolutional Networks for Semantic Segmentation" en 2015	Realiza la segmentación semántica de imágenes en lugar de solo clasificar las imágenes por píxeles individuales [84]
46	U-Net	Propuesto en un artículo de investigación titulado "U-Net: Convolutional Networks for Biomedical Image Segmentation" en 2015	Arquitectura de red neuronal convolucional que se utiliza para la segmentación de imágenes biomédicas con alta precisión.[85]
46	DeepLab	Propuesto en un artículo de investigación titulado "DeepLab: Semantic Image Segmentation with Deep Convolutional	El modelo de red neuronal convolucional profunda se destaca por su velocidad, precisión y simplicidad, y ha demostrado un rendimiento sobresaliente en varias tareas de

		Nets, Atrous Convolution, and Fully Connected CRFs" en 2017	segmentación semántica.[86].
47	MaskLab	Propuesto en un artículo titulado MaskLab: Instance Segmentation by Refining Object Detection with Semantic and Direction Features	Modelo que se enfoca en la combinación de métodos de detección y segmentación para lograr una mejor segmentación de instancias. [87]

Nota: Lista de modelos de aprendizaje profundo

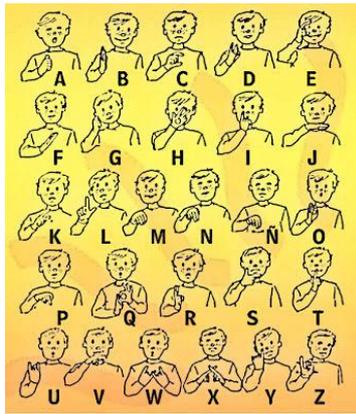
ANEXO 06: LISTA DE LENGUAJE DE SEÑAS

TABLA XII. LISTA DE LENGUAJE DE SEÑAS

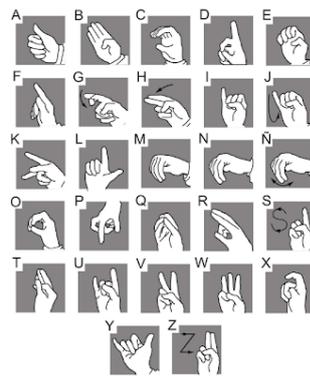
<p>Lengua de Señas americana</p>	<p>Lengua de Señas Británico</p> <p>Standard Manual Alphabet</p> <p>Draw the signs for your name:</p>	<p>Lengua de Señas Griego</p>
<p>Lengua de Señas Alemán</p> <p>Das Fingeralphabet</p>	<p>Lengua de Señas Egipcio</p> <p>ALFABET DACTILOLOGIC EGIPCI</p>	<p>Lengua de Señas Danés</p>
<p>Lengua de Señas Italiana</p>	<p>Lengua de Señas española</p>	<p>Lengua de Señas Brasileña</p>



Lengua de Señas Argentina



Lengua de Señas Colombiana



Lengua de Señas Peruana



Lengua de Señas Venezolana



Lengua de Señas Polono



Lengua de Señas Polono



Lengua de Señas de Portugal

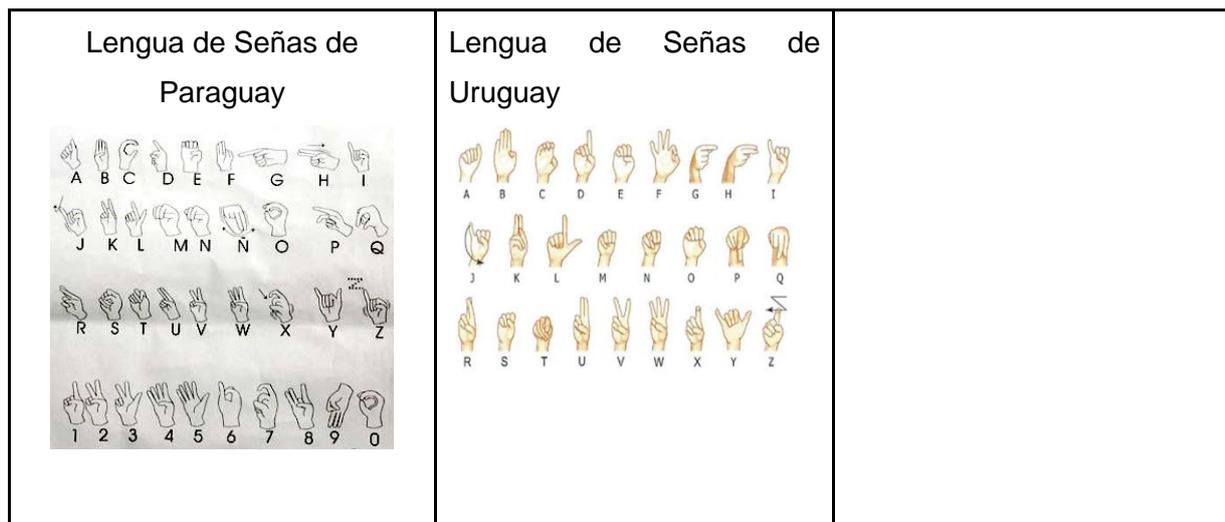


Lengua de Señas de Guatemala



Lengua de Señas de Nicaragua





ANEXO 07: PROCESO DE RECOLECCIÓN DE DATASET

1. Importamos las bibliotecas necesarias:

```
from datetime import timedelta
import cv2
import numpy as np
import os
```

2. Definir la cantidad de fotogramas por segundo que se desean guardar en la variable **SAVING_FRAMES_PER_SECOND**.

```
SAVING_FRAMES_PER_SECOND = 10
```

3. Crear una función llamada **format_timedelta** que acepta un objeto timedelta y devuelve una cadena de texto con el formato adecuado.

```
def format_timedelta(td):
    result = str(td)
    try:
        result, ms = result.split(".")
    except ValueError:
        return result + ".00".replace(":", "-")
    ms = int(ms)
    ms = round(ms / 1e4)
    return f"{result}.{ms:02}".replace(":", "-")
```

4. Crear una función llamada **get_saving_frames_durations** que acepta un objeto cap (de tipo cv2.VideoCapture) y la cantidad de fotogramas por segundo que se desean guardar. Esta función devuelve una lista con las duraciones de los fotogramas que se desean guardar.

```
def get_saving_frames_durations(cap, saving_fps):
    s = []
    clip_duration = cap.get(cv2.CAP_PROP_FRAME_COUNT) / cap.get(cv2.CAP_PROP_FPS)
    for i in np.arange(0, clip_duration, 1 / saving_fps):
        s.append(i)
    return s
```

5. Crear una función llamada main que acepta la ruta del archivo de video. Esta función se encarga de abrir el archivo de video, obtener la cantidad de fotogramas por segundo y la duración de los fotogramas que se desean guardar, crear una carpeta para almacenar los fotogramas y guardar los fotogramas en la carpeta adecuada.

```
def main(video_file):
    filename, _ = os.path.splitext(video_file)
    filename += "-opencv"
    if not os.path.isdir(filename):
        os.mkdir(filename)
    cap = cv2.VideoCapture(video_file)
    fps = cap.get(cv2.CAP_PROP_FPS)
    saving_frames_per_second = min(fps, SAVING_FRAMES_PER_SECOND)
    saving_frames_durations = get_saving_frames_durations(cap, saving_frames_per_second)
    count = 0
    while True:
        is_read, frame = cap.read()
        if not is_read:
            break
        frame_duration = count / fps
        try:
            closest_duration = saving_frames_durations[0]
        except IndexError:
            break
        if frame_duration >= closest_duration:
            frame_duration_formatted = format_timedelta(timedelta(seconds=frame_duration))
            cv2.imwrite(os.path.join(filename, f"frame{frame_duration_formatted}.jpg"), frame)
            try:
                saving_frames_durations.pop(0)
            except IndexError:
                pass
        count += 1
```

6. Verificar si el archivo actual es el archivo principal. Si lo es, llamar a la función main y pasarle la ruta del archivo de video.

```

if __name__ == "__main__":
    import sys
    video_file = sys.argv[1]
    main(video_file)

```

7. Este es un ejemplo de uso:

```

#! /bin/bash
python .\video.py .\videos\piel.mp4

```

ANEXO 08: PROCESO DE PROCESAMIENTO DE LAS IMAGENES

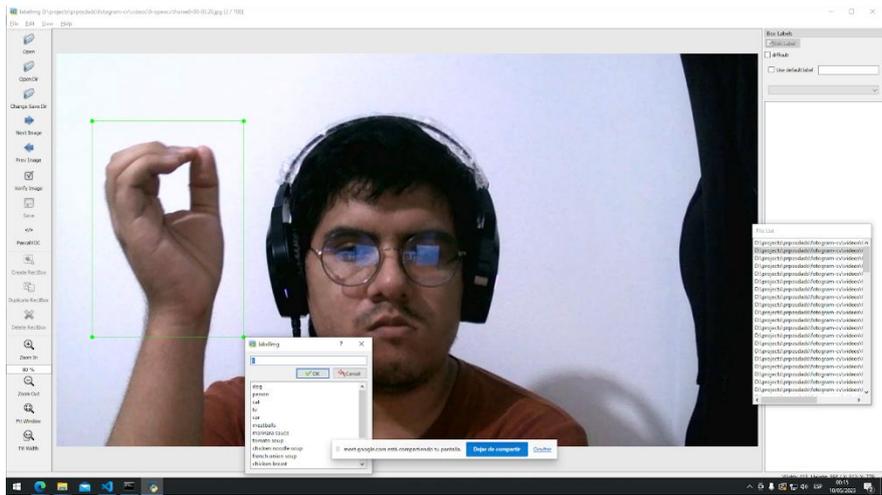


Fig. 32. Procesamiento de número 0 con LabelImg

```

<annotation>
  <folder>collectedimages</folder>
  <filename>a26.jpeg</filename>
  <path>C:\Users\INF\Documents\Tesis\Tesis\SignLanguagePeruvian\Tensorflow\
workspace\images\collectedimages\a26.jpeg</path>
  <source>
    <database>Unknown</database>
  </source>
  <size>
    <width>763</width>
    <height>953</height>
    <depth>3</depth>
  </size>
  <segmented>0</segmented>
  <object>
    <name>A</name>

```

```
<pose>Unspecified</pose>
<truncated>0</truncated>
<difficult>0</difficult>
<bndbox>
  <xmin>22</xmin>
  <ymin>404</ymin>
  <xmax>315</xmax>
  <ymin>629</ymin>
</bndbox>
</object>
</annotation>
```

Fig. 33. Resultado de número 0 en XML

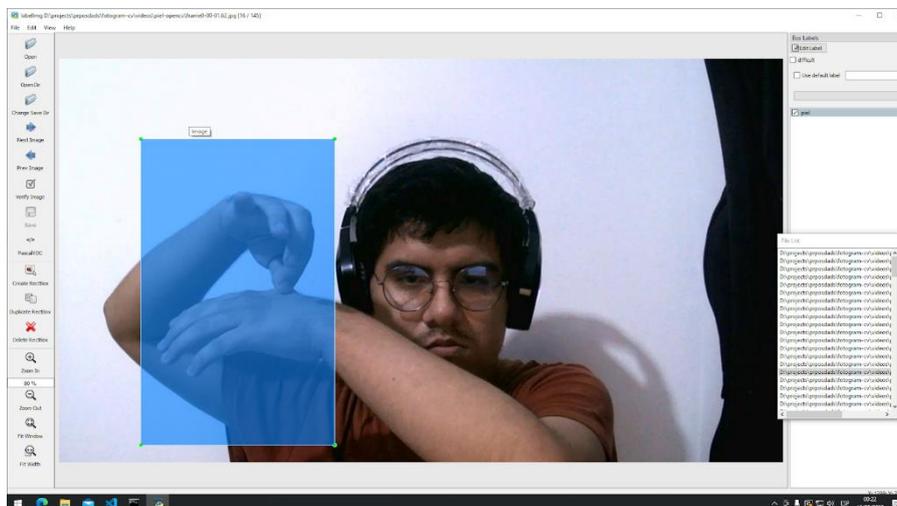


Fig. 34. Procesamiento de la palabra "piel" con LabelImg

Al finalizar este tratamiento de datos, podremos visualizar los xml generado en la carpeta donde guardamos el proyecto.

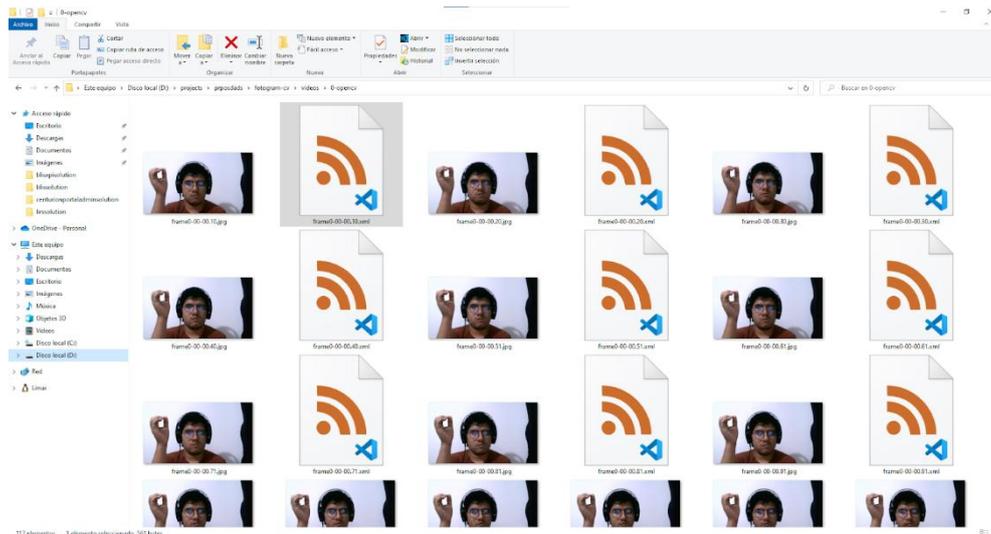


Fig. 35. Muestra de imágenes procesadas por LabelImg

Para poder realizar el tratamiento de los videos para las frases de señas que utiliza gestos y movimientos de las manos, se implementa un algoritmo que permita pasar un vídeo y devuelva como resultado un archivos en formato numpy (.npy) el cual servirá para entrenamiento y test del modelo.

```
import cv2
import numpy as np
from sklearn.model_selection import train_test_split
cap = cv2.VideoCapture('./dataset/piel.mp4')

seq_len = 10
sequences = []
labels = []
sequence = []

while cap.isOpened():
    ret, frame = cap.read()
    if not ret:
        break
    frame = cv2.resize(frame, (640, 480))
    sequence.append(frame)
    if len(sequence) == seq_len:
        label = 'piel'
        labels.append(label)
        sequence = np.stack(sequence, axis=0)
        sequences.append(sequence)
        sequence = []
sequences = np.array(sequences)
labels = np.array(labels)
train_seqs, test_seqs, train_labels, test_labels = train_test_split(sequences, labels, test_size=0.2)
np.save('./prev/train_seqs.npy', train_seqs)
```

```
np.save('./prev/test_seqs.npy', test_seqs)
np.save('./prev/train_labels.npy', train_labels)
np.save('./prev/test_labels.npy', test_labels)
```

Nombre	Fecha de modificación	Tipo	Tamaño
 test_labels.npy	10/05/2023 19:26	Archivo NPY	1 KB
 test_seqs.npy	10/05/2023 19:26	Archivo NPY	13,231 KB
 train_labels.npy	10/05/2023 19:26	Archivo NPY	1 KB
 train_seqs.npy	10/05/2023 19:26	Archivo NPY	49,981 KB

Fig. 36. Resultado de data en formato numpy (.npy)

ANEXO 09: CARTA DE AUTORIZACIÓN PARA LA RECOLECCIÓN DE IMÁGENES

CONSENTIMIENTO INFORMADO PARA LA GRABACIÓN DE VÍDEOS

Por medio de la presente, se otorga su autorización para realizar grabaciones de video en las que usted participará, con el fin de construir un dataset de imágenes de lenguaje de señas.

Al aceptar participar en las grabaciones de video, usted está otorgando su consentimiento para que Santisteban Ostos Andy Josue y Pacheco Contreras Nicolette Isis permita la captura de imágenes de video de las señas y expresiones utilizadas en el lenguaje de señas, con el propósito de recopilar un conjunto de datos que contribuya a la investigación y desarrollo de tecnologías relacionadas con la interpretación y traducción de este idioma. En tal sentido, yo _____ autorizo a que utilice mi imagen con fines de divulgación y promoción, dentro del ámbito de la institución y sus actividades.

Agradecemos su disposición para colaborar en este proyecto y entender que su participación en las grabaciones de video es voluntaria. Queremos asegurarle que se tomarán todas las medidas necesarias para proteger su privacidad y respetar sus derechos como individuo.

Asimismo, Santisteban Ostos Andy Josue y Pacheco Contreras Nicolette Isis se comprometen a no utilizar las grabaciones de vídeo de manera que pueda dañar su reputación o perjudicar su bienestar. También garantizamos que las grabaciones de video no serán vendidas, transferidas ni compartidas con terceros sin su consentimiento previo.

Si en algún momento cambia de opinión y decide revocar su autorización para el uso de las grabaciones de video en las que usted aparece, le solicitamos que nos informe por escrito a la dirección mencionada anteriormente, para que podamos tomar las medidas correspondientes.

DNI:

CONSENTIMIENTO INFORMADO PARA LA GRABACIÓN DE VÍDEOS

Por medio de la presente, se otorga su autorización para realizar grabaciones de video en las que usted participará, con el fin de construir un dataset de imágenes de lenguaje de señas.

Al aceptar participar en las grabaciones de video, usted está otorgando su consentimiento para que Santisteban Ostos Andy Josue y Pacheco Contreras Nicolette Isis permita la captura de imágenes de video de las señas y expresiones utilizadas en el lenguaje de señas, con el propósito de recopilar un conjunto de datos que contribuya a la investigación y desarrollo de tecnologías relacionadas con la interpretación y traducción de este idioma. En tal sentido, yo Felipe Humberto Pacheco Cervantes autorizo a que utilice mi imagen con fines de divulgación y promoción, dentro del ámbito de la institución y sus actividades.

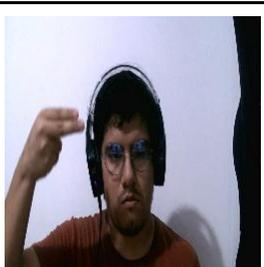
Agradecemos su disposición para colaborar en este proyecto y entender que su participación en las grabaciones de video es voluntaria. Queremos asegurarle que se tomarán todas las medidas necesarias para proteger su privacidad y respetar sus derechos como individuo.

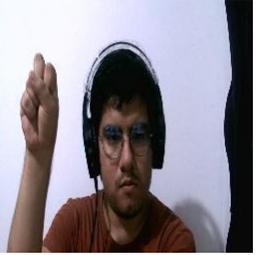
Asimismo, Santisteban Ostos Andy Josue y Pacheco Contreras Nicolette Isis se comprometen a no utilizar las grabaciones de video de manera que pueda dañar su reputación o perjudicar su bienestar. También garantizamos que las grabaciones de video no serán vendidas, transferidas ni compartidas con terceros sin su consentimiento previo.

Si en algún momento cambia de opinión y decide revocar su autorización para el uso de las grabaciones de video en las que usted aparece, le solicitamos que nos informe por escrito a la dirección mencionada anteriormente, para que podamos tomar las medidas correspondientes.


DNI: 77296827

ANEXO 10: IMÁGENES DE LENGUAJE LENGUAJES DE SEÑAS

			
LETRA A	LETRA B	LETRA C	LETRA D
			
LETRA E	LETRA F	LETRA G	LETRA H
			
LETRA I	LETRA K	LETRA L	LETRA M
			
LETRA N	LETRA O		

			
LETRA P	LETRA Q	LETRA R	LETRA S
			
LETRA T	LETRA U	LETRA V	LETRA W
			
LETRA X	LETRA Y		

ANEXO 11: MATRIZ DE CONFUSIÓN DEL MODELO YOLO V5

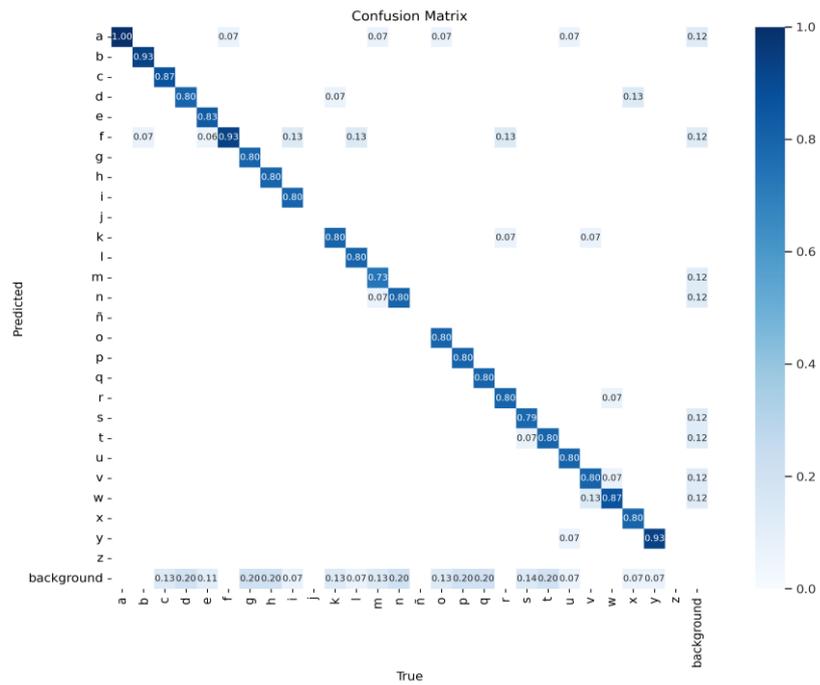


Fig. 37. Gráfico de Matriz de confusión Modelo YOLO V5

ANEXO 12: MATRIZ DE CONFUSIÓN DEL MODELO YOLO V5 Personalizado

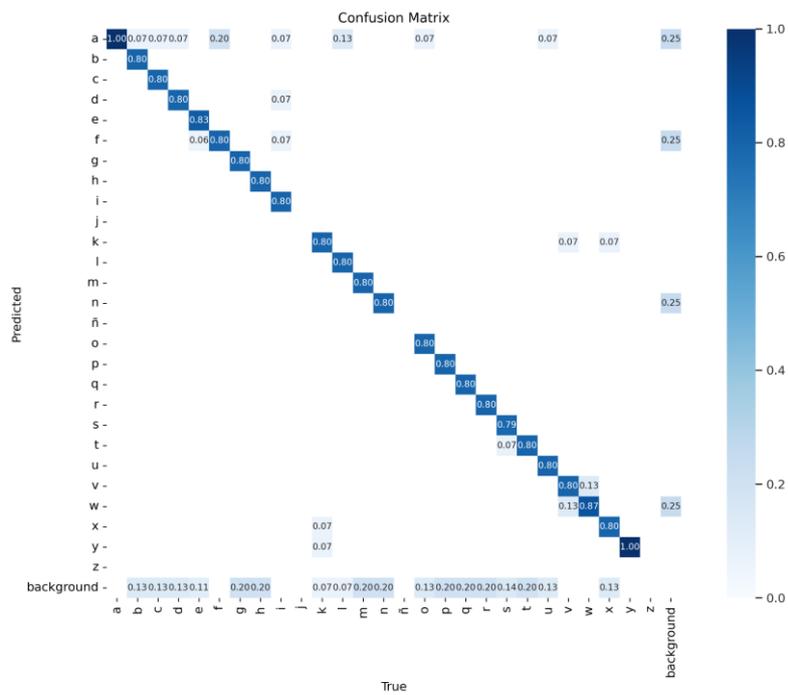


Fig. 38. Gráfico de Matriz de confusión Modelo YOLO V5 personalizado

ANEXO 13: MATRIZ DE CONFUSIÓN DEL MODELO MobileNet V2

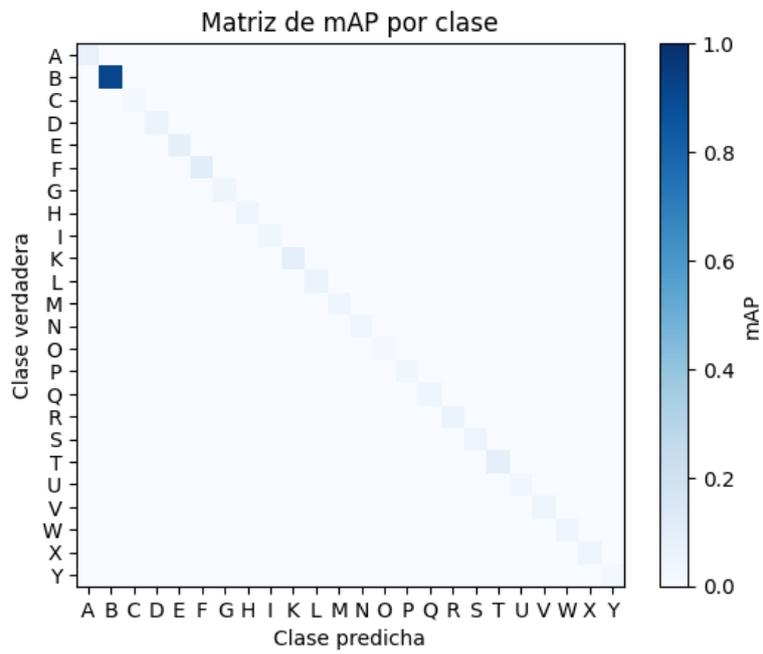


Fig. 39. Gráfico de Matriz de confusión Modelo MobileNet V2