
Señas Chapinas: Traductor de LENSEGUA

Reconocimiento de lengua de señas mediante visión por computadora

Luis Diego Santos Cuéllar



UNIVERSIDAD DEL VALLE DE GUATEMALA

Facultad de Ingeniería



Señas Chapinas: Traductor de LENSEGUÁ

Reconocimiento de lengua de señas mediante visión por computadora

Trabajo de graduación en modalidad de Megaproyecto Tecnológico

presentado por

Luis Diego Santos Cuéllar

Para optar al grado académico de Licenciado en Ingeniería en Ciencias
de la Computación

Guatemala, noviembre del 2024

Vo.Bo.:

(f) _____
Ing. Javier Josue Fong Guzmán

Tribunal Examinador:

(f) _____
Ing. Javier Josue Fong Guzmán

(f) _____

(f) _____

Fecha de aprobación: Guatemala,

Prefacio

En un mundo cada vez más interconectado, la comunicación efectiva se erige como un pilar fundamental para la convivencia y el desarrollo social. Sin embargo, las barreras lingüísticas y comunicativas continúan afectando a millones de personas, especialmente a aquellas con discapacidades auditivas. En este contexto, el presente proyecto, "Señas Chapinas: Traductor de LENSEGUA", surge como una respuesta innovadora para abordar estos desafíos en Guatemala, un país con una rica diversidad cultural y lingüística.

Este trabajo tiene como propósito desarrollar un sistema de visión por computadora que permita el reconocimiento en tiempo real de la lengua de señas de Guatemala (LENSEGUA). A través de la integración de tecnologías avanzadas, como el procesamiento de imágenes y redes neuronales, el proyecto busca no solo facilitar la comunicación diaria entre personas con y sin discapacidad auditiva, sino también promover la inclusión social y mejorar la calidad de vida de aquellos que enfrentan obstáculos comunicativos.

A lo largo de este informe, se detallarán las etapas del desarrollo del sistema, desde la recopilación y preparación de datos hasta el entrenamiento y evaluación del modelo. Asimismo, se reflexionará sobre el impacto potencial de esta herramienta en la vida de los guatemaltecos y en el contexto de las iniciativas inclusivas promovidas por el gobierno. Este trabajo no solo representa un avance tecnológico, sino también un compromiso con la equidad y el reconocimiento de la lengua de señas como un medio legítimo de comunicación.

Agradecimientos

Expreso mi más sincero agradecimiento a todas las personas e instituciones que hicieron posible la realización de este proyecto.

En primer lugar, agradezco a mi asesor, el Ing. Javier Josue Fong Guzmán, por su guía y apoyo incondicional a lo largo de este proceso. Su conocimiento y experiencia fueron fundamentales para el desarrollo de este trabajo, y su aliento constante me motivó a superar los desafíos que se presentaron.

Agradezco también a la Universidad del Valle de Guatemala, por brindarme la oportunidad de formarme en un entorno académico estimulante y por fomentar un espíritu de innovación y compromiso social. La educación recibida en esta institución ha sido clave para llevar a cabo este proyecto y contribuir al desarrollo de soluciones tecnológicas que beneficien a la comunidad.

Finalmente, quiero dedicar un agradecimiento especial a mis padres y a mi hermana, quienes han sido mi mayor apoyo en cada etapa de mi vida. Su aliento incondicional me ha permitido alcanzar mis metas y perseguir mis sueños. Sin su apoyo, este proyecto no habría sido posible.

A todos ustedes, gracias por ser parte de este proyecto y por contribuir a la realización de "Señas Chapinas: Traductor de LENSEGUA".

Índice

Prefacio	v
Agradecimientos	vii
Lista de Figuras	xii
Lista de Cuadros	xiii
Resumen	xv
1. Introducción	1
2. Objetivos	3
2.1. Objetivo General	3
2.2. Objetivos Específicos	3
3. Justificación	5
4. Marco Teórico	7
4.1. Discapacidad auditiva	7
4.1.1. Grados de discapacidad auditiva	7
4.1.2. Impacto en la comunicación interpersonal	8
4.2. Lengua de señas de Guatemala	8
4.2.1. Origen y evolución	8
4.2.2. Características	8
4.2.3. Diferenciación entre señas	8
4.3. Fundamentos de visión por computadora	9
4.3.1. Procesamiento de imágenes	9
4.3.2. Técnicas básicas	9
4.3.3. Herramientas	9
4.4. Aplicaciones de visión por computadora	10
4.4.1. Medicina	10
4.4.2. Industria automotriz	10
4.4.3. Seguridad y vigilancia	10
4.4.4. Manufactura	10
4.5. Redes neuronales	10
4.5.1. Fundamentos	10
4.5.2. Tipos de redes neuronales	11

4.5.3. Aplicaciones en visión por computadora	11
4.6. Evaluación de modelos de visión por computadora	11
4.6.1. Matriz de confusión	11
4.6.2. Sensibilidad	11
4.6.3. Puntuación F1	12
5. Metodología	13
5.1. Delimitación del universo de palabras	13
5.2. Recopilación de datos	14
5.3. Preparación del conjunto de datos	15
5.4. Procesamiento de los videos	16
5.5. Labeling de los datos	17
5.6. Normalización de los datos	17
5.7. Oversampling	18
5.8. Entrenamiento del modelo	18
5.9. Evaluación del modelo	19
5.10. Evaluación del conjunto de datos	20
5.11. Publicación del modelo	22
6. Antecedentes	23
7. Alcance	25
8. Resultados	27
8.1. Conjunto de datos	27
8.1.1. Datos de lengua de señas de Guatemala	27
8.1.2. Análisis del conjunto de datos	27
8.2. Proceso iterativo de desarrollo del modelo	31
8.2.1. Modelo base	31
8.2.2. Aumento de complejidad del modelo base	33
8.2.3. Adición de dropout al modelo base	35
8.2.4. Fine tuning de las capas dropout	37
8.2.5. Adición de normalización por lotes al modelo base	41
8.2.6. Combinación de dropout y normalización por lotes	43
8.2.7. Elección del modelo final	47
8.3. Evaluación del modelo	47
8.3.1. Reconocimiento de una palabra	47
8.3.2. Reconocimiento de múltiples palabras	48
8.4. Aplicaciones del modelo	48
8.4.1. Pruebas en tiempo real	48
8.4.2. Pruebas del API	49
9. Análisis de resultados	51
9.1. Desempeño del modelo	51
9.2. Limitaciones del modelo	52
9.3. Cumplimiento de los objetivos	53
10. Conclusiones	55
11. Recomendaciones	57
Bibliografía	60

Lista de Figuras

4.1. Escala logarítmica de diferentes sonidos cotidianos. Fuente: [7]	7
5.1. Ejemplos de captura de los videos para el conjunto de datos. Fuente: [16]	14
5.2. Ejemplo de reflejo de un video.	15
5.3. Proceso de preparación del conjunto de datos.	16
5.4. Ejemplo de uso de MediaPipe. Fuente: [10]	17
5.5. Ejemplo de datos procesados.	18
5.6. Ejemplo de una red neuronal feedforward sencilla.	19
5.7. Implementación de mejoras en la red neuronal feedforward.	19
5.8. Ejemplo de visualización de datos con PCA.	21
5.9. Ejemplo de matriz de confusión.	22
8.1. Análisis de similitud entre las clases <i>como</i> y <i>agua</i>	28
8.2. Análisis de similitud entre las clases <i>como</i> y <i>beber</i>	28
8.3. Análisis de similitud entre las clases <i>cuando</i> y <i>donde</i>	29
8.4. Análisis de similitud entre las clases <i>sed</i> y <i>quien</i>	29
8.5. Análisis de similitud entre todas las clases	30
8.6. Balance de clases en el conjunto de datos de entrenamiento	30
8.7. Modelo inicial	31
8.8. Historial de entrenamiento del modelo base	32
8.9. Matriz de confusión del modelo base	32
8.10. Modelo con aumento de complejidad	33
8.11. Historial de entrenamiento del modelo con aumento de complejidad	34
8.12. Matriz de confusión del modelo con aumento de complejidad	34
8.13. Modelo base con dropout	35
8.14. Historial de entrenamiento del modelo base con dropout	36
8.15. Matriz de confusión del modelo base con dropout	36
8.16. Primera iteración de fine tuning de la tasa de dropout	37
8.17. Historial de entrenamiento del primer modelo de fine tuning de la tasa de dropout .	38
8.18. Matriz de confusión del primer modelo de fine tuning de la tasa de dropout	38
8.19. Segunda iteración de fine tuning de la tasa de dropout	39
8.20. Historial de entrenamiento del segundo modelo de fine tuning de la tasa de dropout	40
8.21. Matriz de confusión del segundo modelo de fine tuning de la tasa de dropout	40
8.22. Modelo con normalización por lotes	41
8.23. Historial de entrenamiento del modelo con normalización por lotes	42
8.24. Matriz de confusión del modelo con normalización por lotes	42
8.25. Primer modelo con dropout y normalización por lotes	43

8.26. Historial de entrenamiento del primer modelo con dropout y normalización por lotes	44
8.27. Matriz de confusión del primer modelo con dropout y normalización por lotes	44
8.28. Segundo modelo con dropout y normalización por lotes	45
8.29. Historial de entrenamiento del segundo modelo con dropout y normalización por lotes	46
8.30. Matriz de confusión del segundo modelo con dropout y normalización por lotes	46
8.31. Reconocimiento de lengua de señas de Guatemala en tiempo real (una palabra)	47
8.32. Reconocimiento de lengua de señas de Guatemala en tiempo real (múltiples palabras)	48
8.33. Reconocimiento de la palabra <i>ayer</i> en tiempo real	49
8.34. Reconocimiento de la palabra <i>comer</i> en tiempo real	49
8.35. Reconocimiento de la palabra <i>universidad</i> utilizando el API	50
8.36. Reconocimiento de la palabra <i>tu</i> utilizando el API	50

Lista de Cuadros

8.1. Desempeño del modelo base	32
8.2. Desempeño del modelo con aumento de complejidad	34
8.3. Desempeño del modelo base con dropout	36
8.4. Desempeño del primer modelo de fine tuning de la tasa de dropout	38
8.5. Desempeño del segundo modelo de fine tuning de la tasa de dropout	40
8.6. Desempeño del modelo con normalización por lotes	42
8.7. Desempeño del primer modelo con dropout y normalización por lotes	44
8.8. Desempeño del segundo modelo con dropout y normalización por lotes	46
8.9. Desempeño del segundo modelo de la combinación de dropout y normalización por lotes	47

Resumen

El proyecto "Señas Chapinas: Traductor de LENSEGUA" tiene como objetivo desarrollar un sistema de visión por computadora destinado al reconocimiento de la lengua de señas de Guatemala (LENSEGUA). En un contexto donde las barreras comunicativas afectan a la población con discapacidad auditiva, esta iniciativa busca facilitar la interacción diaria entre personas oyentes y no oyentes, promoviendo la inclusión social y mejorando la calidad de vida de quienes enfrentan desafíos comunicativos.

El desarrollo del sistema se llevó a cabo a través de una metodología estructurada en varias etapas, que incluyó la delimitación de un universo de palabras de uso cotidiano, la recopilación y procesamiento de datos mediante videos de señas, y el entrenamiento de modelos de redes neuronales. Se utilizaron técnicas avanzadas de procesamiento de imágenes, específicamente la librería MediaPipe de Google, para detectar y seguir los movimientos de las manos en tiempo real.

El modelo se entrenó con un conjunto de datos que incluye múltiples tomas desde diferentes ángulos, lo que permite una mayor precisión en diversas condiciones de visualización. Además, se implementaron métodos de validación para asegurar el correcto reconocimiento de las señas, con el fin de optimizar su rendimiento.

Este proyecto no solo se centra en la creación de un traductor funcional, sino que también busca ser una herramienta accesible que permita el uso de LENSEGUA desde dispositivos móviles, contribuyendo así a eliminar las barreras de comunicación en la sociedad guatemalteca. En resumen, "Señas Chapinas" representa un avance significativo en la promoción de la equidad comunicativa y la inclusión de personas con discapacidad auditiva en Guatemala.

CAPÍTULO 1

Introducción

En un mundo donde la comunicación es fundamental, las barreras lingüísticas y comunicativas pueden crear divisiones significativas. El proyecto "Señas Chapinas: Traductor de LENSEGUA" surge como una respuesta innovadora para eliminar estas barreras entre los guatemaltecos, proporcionando un traductor de lengua de señas de Guatemala accesible desde dispositivos móviles. Esta iniciativa no solo busca facilitar la interacción diaria, sino que también promueve la inclusión y mejora la calidad de vida de quienes enfrentan desafíos comunicativos a diario.

Con el propósito de alcanzar su objetivo principal, el proyecto establece una serie de metas específicas. Estas incluyen la investigación exhaustiva de tecnologías disponibles, la exploración de conjuntos de datos para el entrenamiento del reconocimiento de señas y la continua mejora del algoritmo de reconocimiento para garantizar su precisión en tiempo real. Asimismo, se plantea la definición de los requisitos de infraestructura necesarios para ejecutar el reconocimiento de lengua de señas en un servidor en la nube, con el fin de hacer esta herramienta accesible para una amplia audiencia. El producto final de este proyecto será un sistema de visión por computadora para la detección y reconocimiento de la lengua de señas de Guatemala.

Con el fin de cumplir con los objetivos generales y específicos del proyecto, se define una metodología que cubre todos los aspectos del proyecto, desde la recopilación de datos hasta el entrenamiento y validación del modelo. Como primer paso de esta metodología, y esencial para el proyecto, será la delimitación del universo de palabras, ya que esto define el alcance del proyecto y tendrá un gran impacto en el desempeño del modelo. Para completar estas tareas, se utilizarán librerías como MediaPipe de Google, la cual permite realizar el reconocimiento de manos en tiempo real. Durante el proceso de entrenamiento del modelo, se aplicará un proceso de mejora continua, buscando el desempeño óptimo del mismo.

En un contexto donde el gobierno guatemalteco muestra interés en promover iniciativas inclusivas, el proyecto "Señas Chapinas: Traductor de LENSEGUA" se alinea con estos esfuerzos al facilitar el aprendizaje y uso de la lengua de señas en la sociedad. Con su enfoque en la equidad comunicativa, este proyecto se posiciona como una herramienta fundamental para fomentar la integración y la igualdad de oportunidades en la comunicación para todos los ciudadanos guatemaltecos.

CAPÍTULO 2

Objetivos

2.1. Objetivo General

Desarrollar un sistema de visión por computadora para el reconocimiento de la lengua de señas de Guatemala. Este proyecto busca impulsar a Guatemala a un futuro cercano en el cual se puedan eliminar barreras de comunicación para los guatemaltecos con discapacidad auditiva.

2.2. Objetivos Específicos

- Crear un conjunto de datos compuesto por al menos veinticinco palabras de la lengua de señas de Guatemala, utilizando palabras que puedan formar una gran cantidad de frases de uso cotidiano.
- Identificar debidamente los videos que forman parte del conjunto de datos, para que se puedan utilizar para entrenar el modelo.
- Diseñar un sistema de visión por computadora capaz de identificar en tiempo real puntos clave en las extremidades de una persona, con el objetivo de extraer información que se utilizará para identificar palabras del conjunto de datos.
- Diseñar un sistema de inteligencia artificial para la identificación de las palabras seleccionadas de la lengua de señas de Guatemala. Este sistema debe utilizar la información provista por el sistema de visión por computadora para identificar las palabras del conjunto de datos en tiempo real.
- Diseñar un sistema de inteligencia artificial capaz de identificar el momento en el que el usuario termina una seña y empieza con la siguiente, con el objetivo de poder identificar las señas de manera continua.

CAPÍTULO 3

Justificación

Los humanos son seres sociales, y toda nuestra sociedad está modelada en torno a este hecho irrefutable. El modo de vida cotidiano está diseñado alrededor de la comunicación con otras personas. La educación, el trabajo, y muchas otras actividades cotidianas dependen fuertemente de nuestras habilidades de comunicación. No obstante, no todos tienen acceso a estas oportunidades de comunicación, debido a barreras que persisten en nuestra sociedad. Estas barreras nacen de las diferencias en cómo nos comunicamos, y presentan un gran obstáculo que se debería trabajar como sociedad para eliminar.

De esta problemática nace el proyecto “Señas Chapinas: Traductor de LENSEGUÁ”, el cual busca eliminar estas barreras de comunicación entre guatemaltecos. Eliminar estas barreras implica finalmente dar igual acceso a todos a las oportunidades para comunicarse con otras personas. Para muchos, esto puede parecer un problema insignificante, pero para las personas que deben lidiar con esta problemática todos los días de su vida, esto implica un cambio radical en la forma en la que se comunican.

En su esencia, el proyecto “Señas Chapinas: Traductor de LENSEGUÁ” es un traductor de lengua de señas de Guatemala que puede ser utilizado desde cualquier dispositivo móvil. Para el usuario objetivo, será mucho más que solo un traductor, ya que representa una solución a muchos de sus problemas cotidianos. Eliminar estas barreras de comunicación en la vida diaria de los usuarios, implicará una mejora significativa en su calidad de vida.

Adicionalmente, es importante considerar la actualidad del país, ya que el gobierno ha mostrado compromiso por promover iniciativas de difusión de la lengua de señas de Guatemala. Esto se evidencia con el acuerdo gubernativo 121-2021 [5], el cual establece que el gobierno, junto a CONADI y el Ministerio de Educación, promoverán el aprendizaje de la lengua de señas. Tomando en cuenta iniciativas como esta, una aplicación como “Señas Chapinas: Traductor de LENSEGUÁ” podría ser de gran ayuda para promover el uso y aprendizaje de la lengua de señas de Guatemala.

En resumen, este proyecto emerge como una herramienta crucial para derribar las barreras comunicativas que separan a los guatemaltecos, ofreciendo un acceso equitativo a la comunicación a través de su innovador traductor de lengua de señas de Guatemala. Esta solución no solo facilita la interacción diaria, sino que representa un avance significativo hacia la inclusión y la mejora de la calidad de vida de quienes se enfrentan a obstáculos comunicativos a diario.

CAPÍTULO 4

Marco Teórico

4.1. Discapacidad auditiva

4.1.1. Grados de discapacidad auditiva

La discapacidad auditiva abarca desde una pérdida parcial hasta una pérdida total de la capacidad para escuchar, y se categoriza en distintos grados según el umbral auditivo de la persona. Se considera pérdida auditiva leve cuando el nivel de perdida está entre 20 y 40 dB, moderada si está en el rango de 41 a 60 dB, severa cuando está entre 61 a 80 dB, y profunda si supera los 81 dB [21]. Esta clasificación nos ayuda a comprender la diversidad de experiencias de las personas con discapacidad auditiva, desde una ligera dificultad para escuchar hasta la sordera total.

La escala de decibeles (dB) utilizada para medir la pérdida auditiva se basa en una medición logarítmica que refleja la sensibilidad del oído humano a diferentes niveles de sonido. En esta escala, un aumento de 10 dB indica que el volumen percibido de un sonido se duplicó [4]. Por ejemplo, una diferencia de 20 dB significa que un sonido es cuatro veces más fuerte que el sonido al cual se está comparando. Comparar los niveles de pérdida auditiva con el volumen de algunos ruidos cotidianos sirve como punto de referencia de qué implica cada grado de discapacidad auditiva.



Figura 4.1: Escala logarítmica de diferentes sonidos cotidianos. Fuente: [7]

4.1.2. Impacto en la comunicación interpersonal

La pérdida auditiva puede afectar la capacidad de comunicación de un individuo, al generar interferencias significativas en su habilidad para comprender el habla y otros sonidos presentes en su entorno. Estos obstáculos en la comunicación interpersonal pueden llegar a tener un gran impacto en la vida diaria de las personas afectadas. Las personas con una pérdida auditiva severa o profunda a menudo utilizan métodos alternativos de comunicación, como la lectura de labios o la lengua de señas, para interactuar con otras personas. Sin embargo, estos métodos alternativos no son una solución perfecta, y a menudo pueden llevar a dificultades en la comunicación interpersonal. Esta dificultad en la comunicación puede llevar al aislamiento social, obstáculos en la educación y en el empleo, e incluso a la disminución en la calidad de vida de los individuos afectados.

La Biblioteca Nacional de Medicina [19] advierte sobre el impacto negativo que la falta de acceso a la comunicación efectiva puede tener en la vida de las personas con discapacidad auditiva. Este artículo hace particular énfasis en las dificultades que se pueden presentar tanto en la educación como en el empleo. En la educación, las dificultades adicionales que enfrenta un estudiante con discapacidad auditiva pueden llevar a un desempeño académico reducido y, en algunos casos, incluso al abandono de los estudios. Mientras que la tasa de abandono a nivel de secundaria es del 19 % para la población general, este valor se dispara a 44 % para estudiantes con un nivel severo o profundo de discapacidad auditiva [3]. Estas dificultades continúan en el ámbito laboral, ya que muchas empresas se oponen a contratar personas con discapacidades auditivas.

4.2. Lengua de señas de Guatemala

4.2.1. Origen y evolución

La lengua de señas de Guatemala (LENSEGUA) es la lengua principal utilizada por la comunidad con discapacidad auditiva en Guatemala, representando un medio de comunicación importante para este grupo de personas en el país. Este lenguaje nace para satisfacer las necesidades comunicativas de las personas con discapacidad auditiva del país, y ha evolucionado con el tiempo para incluir un amplio vocabulario y una gramática propia. Según el Congreso de la República de Guatemala, LENSEGUA fue reconocida oficialmente en el 2020 [12]. Desde su reconocimiento oficial, se ha impulsado activamente el uso y la enseñanza de LENSEGUA en diversas instituciones educativas y comunidades de Guatemala. El objetivo de esto ha sido fomentar su difusión y valoración como una forma legítima de comunicación. La integración y promoción de esta lengua de señas en el ámbito educativo y social no solo fortalece la identidad y la inclusión de la comunidad con discapacidad auditiva, sino que también enriquece la diversidad lingüística y cultural del país.

4.2.2. Características

LENSEGUA es un idioma visual-gestual, lo que significa que se comunica a través de movimientos de las manos, expresiones faciales y el uso del espacio corporal. Las características de LENSEGUA incluyen una estructura gramatical única que difiere del español hablado. Por ejemplo, el orden de las palabras en una oración en LENSEGUA puede diferir significativamente del orden en español.

4.2.3. Diferenciación entre señas

La diferenciación entre señas es un aspecto crítico de este idioma, ya que pequeñas variaciones en el movimiento de las manos o la expresión facial pueden cambiar completamente el significado de

una seña. En la lengua de señas de Guatemala, pequeñas variaciones como inclinarse hacia adelante pueden cambiar el significado de la frase. Es importante tomar en cuenta estas complejidades de LENSEGUA para poder identificar correctamente el significado de las palabras y las frases.

4.3. Fundamentos de visión por computadora

4.3.1. Procesamiento de imágenes

La visión por computadora es un campo de la inteligencia artificial que permite a las computadoras interpretar y comprender el contenido de las imágenes y videos. Este campo se basa en la adquisición, procesamiento y análisis de imágenes digitales para extraer información significativa. Los objetivos de la visión por computadora incluyen la automatización de tareas que requieren procesamiento visual, como el reconocimiento de objetos, la detección de patrones y el seguimiento de movimientos [6].

4.3.2. Técnicas básicas

Los principios básicos de la visión por computadora incluyen técnicas de procesamiento de imágenes, como la mejora de imágenes, la segmentación de imágenes y la extracción de características. La mejora de imágenes puede incluir el ajuste de contraste y eliminación de ruido. La segmentación de imágenes se enfoca en la división de una imagen en partes significativas, con el objetivo de descartar información que no sea útil. Por último, la extracción de características se enfoca en la identificación de elementos clave dentro de una imagen, como bordes o puntos de interés. Estas técnicas son fundamentales para el desarrollo de aplicaciones que requieren una comprensión detallada de las imágenes [1].

4.3.3. Herramientas

En el ámbito de la visión por computadora, existen varias herramientas y bibliotecas de software que facilitan la implementación de técnicas de procesamiento de imágenes y análisis visual. Entre las más utilizadas se encuentran OpenCV y MediaPipe.

Open Source Computer Vision Library (OpenCV) es una biblioteca de software libre de visión artificial y aprendizaje automático. Esta proporciona más de 2500 algoritmos optimizados para realizar una amplia gama de tareas, como la detección y reconocimiento de rostros, la identificación de objetos, la clasificación de acciones en videos, el seguimiento de movimientos y la reconstrucción de estructuras 3D [20].

Desarrollado por Google, MediaPipe es un marco multiplataforma para la construcción de aplicaciones multimedia. MediaPipe proporciona soluciones de vanguardia para la detección y seguimiento de manos, detección de rostros, entre otras. Es particularmente útil para aplicaciones que requieren el seguimiento en tiempo real y la interacción basada en gestos [11].

Algo que estas dos herramientas tienen en común es el lenguaje de programación que se utiliza, ya que ambas son librerías de Python. Python es un lenguaje de programación ampliamente utilizado en visión por computadora, especialmente cuando se combina con bibliotecas como OpenCV y MediaPipe. Estas herramientas permiten a los desarrolladores implementar aplicaciones avanzadas de visión por computadora de manera eficiente y con buenos resultados. La principal ventaja de utilizar estas librerías es que no se necesita entrenar los modelos de visión por computadora desde

cero, ya que en muchos casos se puede utilizar uno de los modelos que forman parte de estas librerías [2].

4.4. Aplicaciones de visión por computadora

4.4.1. Medicina

La visión por computadora tiene una amplia gama de aplicaciones en diversos campos. En la medicina, se utiliza para el diagnóstico asistido por computadora, ayudando a los médicos a detectar enfermedades a partir de imágenes médicas como radiografías y resonancias magnéticas. Adicionalmente, puede ayudar a que los diagnósticos sean más certeros, ya que la tecnología puede servir como una segunda opinión del diagnóstico [8].

4.4.2. Industria automotriz

En el sector automotriz, la visión por computadora es fundamental para el desarrollo de vehículos autónomos y sistemas avanzados de asistencia al conductor. Estos sistemas dependen de la capacidad de los vehículos para identificar y responder a señales de tráfico, peatones y otros obstáculos en la carretera.

4.4.3. Seguridad y vigilancia

En el ámbito de la seguridad y vigilancia, la visión por computadora se utiliza para la detección de intrusos, el reconocimiento facial y la identificación de comportamientos sospechosos. Estas tecnologías son fundamentales para la prevención de crímenes y la protección de la seguridad pública.

4.4.4. Manufactura

En el ámbito industrial, la visión por computadora se emplea para el control de calidad y la inspección automatizada de productos. Esto puede simplificar de manera drástica los procesos de manufactura, y reduce dramáticamente el tiempo requerido en procesos de control de calidad de los productos.

4.5. Redes neuronales

4.5.1. Fundamentos

Las redes neuronales son un componente esencial de la inteligencia artificial. El funcionamiento de estas se inspira en el funcionamiento del cerebro humano, con una gran cantidad de neuronas. Estas redes están compuestas por capas de nodos, que son como neuronas artificiales, que procesan la información a través de conexiones ponderadas [9].

4.5.2. Tipos de redes neuronales

Existen varios tipos de redes neuronales, cada una adecuada para diferentes tareas. Las redes neuronales feedforward (FNN) son las más simples, donde la información se propaga en una sola dirección, de la entrada a la salida. Las redes neuronales convolucionales (CNN) son especialmente efectivas para el procesamiento de imágenes debido a su capacidad para reconocer patrones espaciales jerárquicos. Por último, las redes neuronales recurrentes (RNN) son adecuadas para el procesamiento de datos secuenciales, como el reconocimiento de voz o la traducción automática [13].

4.5.3. Aplicaciones en visión por computadora

Las aplicaciones de redes neuronales en visión por computadora incluyen la clasificación de imágenes, donde las CNN pueden identificar y categorizar objetos dentro de una imagen, y la detección de objetos, donde se localizan y etiquetan múltiples objetos dentro de una escena. Estas tecnologías son fundamentales para el desarrollo de sistemas avanzados de reconocimiento de señas, como el proyecto "Señas Chapinas: Traductor de LENSEGUA", que utiliza redes neuronales para interpretar el lenguaje de señas guatemalteco en tiempo real.

4.6. Evaluación de modelos de visión por computadora

4.6.1. Matriz de confusión

La matriz de confusión es una herramienta que permite visualizar el rendimiento de un modelo de clasificación. Se organiza en una tabla que muestra las predicciones del modelo en comparación con las verdaderas etiquetas de los datos. Esta matriz se compone de cuatro componentes principales:

- Verdaderos Positivos (TP): Casos correctamente clasificados como positivos.
- Falsos Positivos (FP): Casos incorrectamente clasificados como positivos.
- Verdaderos Negativos (TN): Casos correctamente clasificados como negativos.
- Falsos Negativos (FN): Casos incorrectamente clasificados como negativos.

La matriz de confusión es una herramienta útil para evaluar el rendimiento de un modelo de clasificación, ya que proporciona información detallada sobre los errores del modelo. [14] Adicionalmente, la matriz de confusión se puede utilizar para calcular otras métricas de rendimiento del modelo, como la sensibilidad y la puntuación F1.

4.6.2. Sensibilidad

La sensibilidad, también conocida como recall, es una métrica que mide la proporción de casos positivos que fueron correctamente identificados por el modelo. Un alto valor de sensibilidad indica que el modelo tiene una buena capacidad para detectar la clase positiva, lo cual es especialmente importante en aplicaciones donde se desea minimizar los falsos negativos. [15] Se calcula utilizando la siguiente fórmula:

$$\text{Sensibilidad} = \frac{TP}{TP + FN} \quad (4.1)$$

Donde TP son los verdaderos positivos y FN son los falsos negativos.

4.6.3. Puntuación F1

La puntuación F1, más comúnmente conocida como F1-score, es una métrica que combina la precisión y la sensibilidad en un solo valor. Se utiliza para evaluar el equilibrio entre la capacidad del modelo para identificar casos positivos (sensibilidad) y la exactitud de esas identificaciones (precisión). Se calcula de la siguiente manera:

$$\text{F1-score} = 2 \times \frac{\text{Precisión} \times \text{Sensibilidad}}{\text{Precisión} + \text{Sensibilidad}} \quad (4.2)$$

Donde la precisión se calcula como:

$$\text{Precisión} = \frac{TP}{TP + FP} \quad (4.3)$$

Donde TP son los verdaderos positivos y FP son los falsos positivos.

La puntuación F1 es especialmente útil en situaciones donde hay un desbalance en las clases, ya que proporciona una medida más completa del rendimiento del modelo que la precisión o la sensibilidad por separado. [17] Esta métrica es ampliamente utilizada en la evaluación de modelos de clasificación.

CAPÍTULO 5

Métodología

Con el fin de cumplir con los objetivos generales y específicos del proyecto, se definió una metodología compuesta por seis etapas, cada una con objetivos claramente definidos. La primera de estas consistió en delimitar el universo de palabras que se utilizaría para entrenar el modelo, buscando palabras de uso cotidiano que permitieran crear una gran cantidad de frases con un alto nivel de utilidad. Una vez definido el universo de palabras, se procedió con la recopilación de datos, la cual incluyó crear un conjunto de datos con videos de las señas correspondientes a las diferentes palabras que se utilizarían. El siguiente paso consistió en preparar el conjunto de datos con los videos recopilados anteriormente, lo cual mayormente consistió en identificar manualmente cada señal para que estos datos pudieran ser utilizados en el proceso de entrenamiento. Al tener listo el conjunto de videos, se pudo proceder al procesamiento de las imágenes, el cual se realizó con la librería Mediapipe de Google, que creó esqueletos de los movimientos capturados en los diferentes videos. Este esqueleto se utilizó para entrenar una red neuronal que fuera capaz de identificar las diferentes señas del universo de palabras seleccionado. Como último paso, se evaluó el desempeño del modelo, buscando áreas de mejora.

5.1. Delimitación del universo de palabras

Dada la restricción de tiempo para la finalización del proyecto, resultó esencial establecer un conjunto definido de palabras que se emplearían en el entrenamiento y la evaluación del modelo. La selección de las palabras que formaron parte de este universo fue extremadamente importante para la utilidad del proyecto, ya que elegir las palabras equivocadas pudo haber significado que el proyecto no cumpliera con los objetivos establecidos. Para este proceso, se seleccionaron al menos veinticinco palabras que en conjunto se pudieran utilizar para crear una gran cantidad de frases de uso cotidiano. La primera fase de este proceso requirió de alguien con amplio conocimiento de la lengua de señas de Guatemala, y consistió en identificar al menos cincuenta palabras que se utilizaran con frecuencia. Al tener este dato inicial, se analizó cuáles de estas palabras se podrían utilizar para formar oraciones congruentes, haciendo énfasis en la utilidad y frecuencia de uso de estas. Adicionalmente, se tomó en cuenta el grado de similitud de la señal con las demás palabras del universo de palabras. Esto con el objetivo de reducir la probabilidad de que el modelo confundiera dos señas diferentes debido a que eran muy similares.

Al finalizar la lista original de palabras, se buscó retroalimentación de las personas que ayudaron

con la grabación de los videos de entrenamiento. Este paso fue crucial para asegurar que las palabras seleccionadas fueran relevantes y útiles en el contexto de la lengua de señas de Guatemala. Las opiniones de los colaboradores permitieron identificar posibles mejoras en la selección de palabras y garantizar que el conjunto final reflejara adecuadamente las necesidades de los usuarios. Tomando en cuenta la retroalimentación recibida, se ajustó por una última vez la lista de palabras para asegurar que el universo de palabras seleccionado fuera el más adecuado para el proyecto.

5.2. Recopilación de datos

Al tener un universo de palabras claramente definido, se pudo proceder a recopilar los datos necesarios para representar todas las palabras elegidas. En esta fase del proyecto, nuevamente se requirió del apoyo de una persona con un profundo conocimiento de la lengua de señas de Guatemala para crear los videos que servirían como base del conjunto de datos. Para que el modelo contara con una amplia variedad de datos de entrenamiento, se realizaron quince tomas de cada seña que formara parte del universo de palabras. Se utilizaron tres ángulos de cámara diferentes, de frente y vista en diagonal de cada lado, realizando cinco tomas desde cada uno de los ángulos. El objetivo de utilizar diferentes ángulos fue mejorar la habilidad del modelo para reconocer las señas en condiciones no ideales, ya que no se podía esperar que el usuario siempre estuviera perfectamente alineado a la cámara. De igual manera, el objetivo de realizar cinco tomas desde cada uno de los ángulos fue que el modelo pudiera identificar las señas correctamente a pesar de las ligeras variaciones que se pudieran presentar en los movimientos.

Al tomar estos videos, se buscó que el usuario que realizara las señas lo hiciera de forma clara y precisa, evitando movimientos innecesarios que pudieran confundir al modelo. Adicionalmente, se tuvo cuidado de que el fondo de los videos fuera lo más simple posible, para asegurar que MediaPipe pudiera identificar correctamente las manos del usuario. Durante esta fase del proyecto se realizó el hallazgo de que las personas surdas realizan las señas de forma diferente, por lo que se tuvo que tomar en cuenta este factor al momento de grabar los videos.

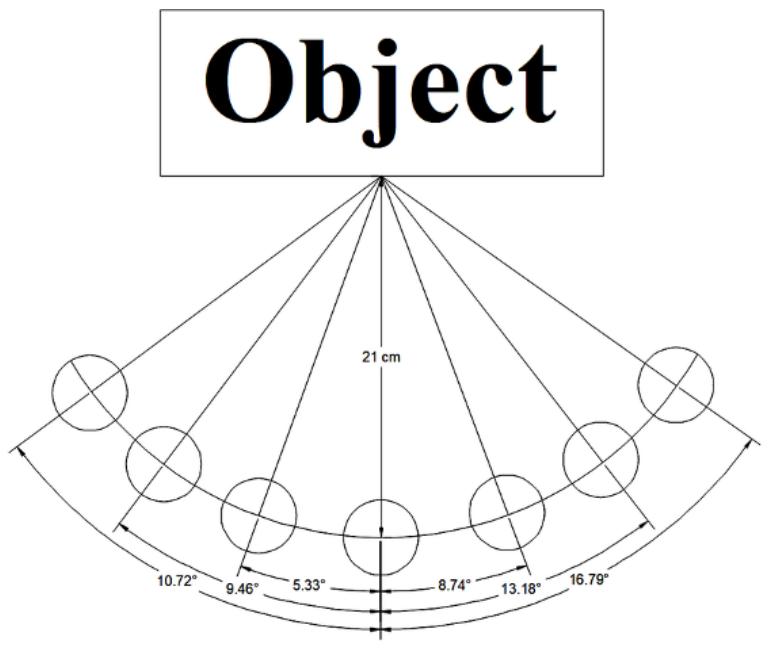


Figura 5.1: Ejemplos de captura de los videos para el conjunto de datos. Fuente: [16]

5.3. Preparación del conjunto de datos

Después de recopilar los datos que formaron parte del conjunto de datos, se procedió con la creación de este. El primer paso para preparar el conjunto de datos fue identificar correctamente todos los videos que se tomaron en el paso anterior. Para esto, nuevamente se requirió de la asistencia de alguien con conocimiento de la lengua de señas de Guatemala, ya que cualquier error llevaría a que el modelo identificara erróneamente las señas. La identificación de los videos se realizó por medio del nombre del archivo, el cual siguió el siguiente formato: Seña-Vista-Toma. El valor de “vista” fue *Frente*, *Der*, o *Izq*, para indicar si el video se tomó con una vista frontal, o una vista lateral desde la izquierda o la derecha, el valor de “seña” indicó la palabra a la que corresponde el video, y el valor de “toma” indicó el número de toma que se realizó. Combinado con el valor de “toma”, se pudieron distinguir claramente los videos entre sí, lo cual resultó útil si el modelo terminó teniendo problemas para identificar la seña en un video en específico.

Tomando en cuenta el hallazgo de que las personas surdas realizan las señas de forma diferente, se tuvo que tomar en cuenta este factor al momento de preparar el conjunto de datos. Para garantizar que el modelo pudiera identificar las señas correctamente, se reflejaron los videos sobre el eje vertical, de forma que las señas realizadas por una persona zurda se vieran como si fueran realizadas por una persona diestra, y viceversa. Esto permitió que el modelo pudiera identificar las señas de forma correcta sin importar si el usuario era zurdo o diestro.

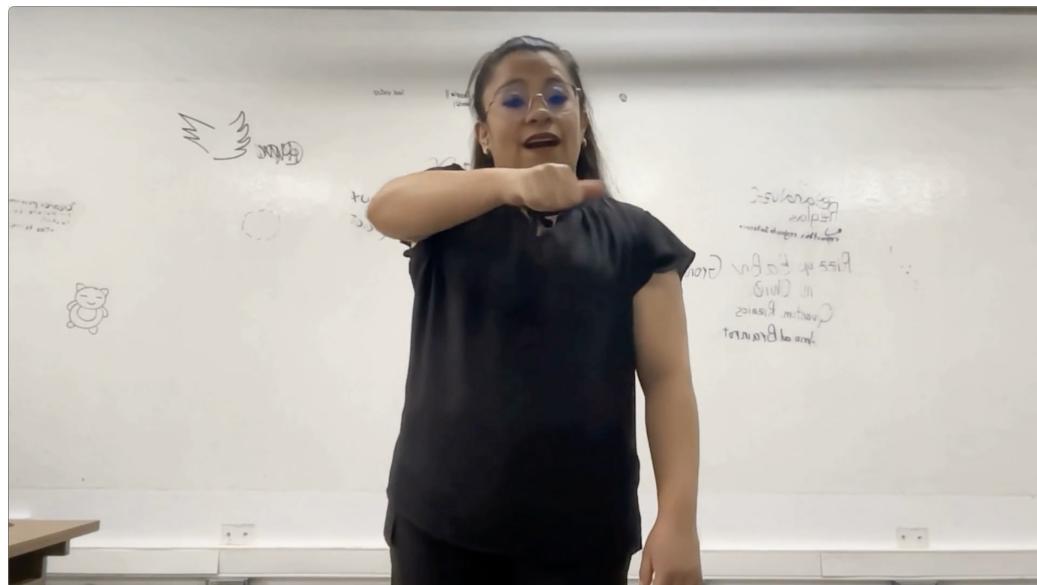


Figura 5.2: Ejemplo de reflejo de un video.

5.4. Procesamiento de los videos



Figura 5.3: Proceso de preparación del conjunto de datos.

Con el conjunto de datos ya definido, se pudo procesar los videos para crear datos que resultaran útiles para entrenar y validar el modelo. Para esto, se utilizó la librería MediaPipe de Google, la cual permite realizar el reconocimiento de manos en tiempo real. Esta librería utiliza modelos de aprendizaje automático para detectar y seguir la posición de las manos en imágenes o videos, ofreciendo información detallada sobre la mano, incluyendo la detección de puntos clave y gestos. Utilizando esta librería, se crearon esqueletos que representaron los movimientos capturados en cada video.

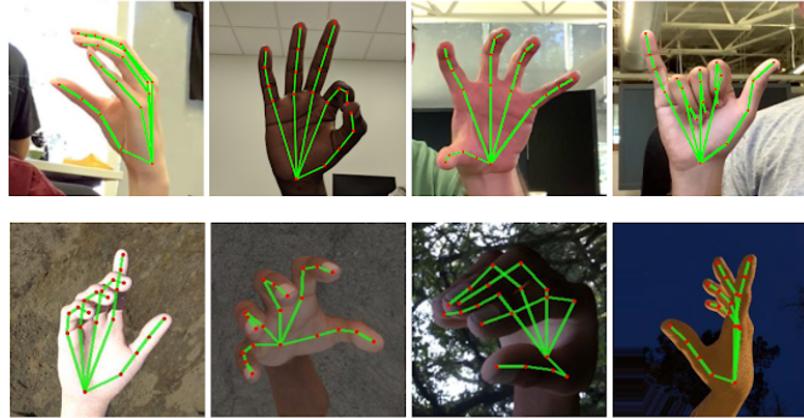


Figura 5.4: Ejemplo de uso de MediaPipe. Fuente: [10]

Utilizando estos esqueletos, se creó un conjunto de datos con todas las coordenadas relevantes para poder entrenar el modelo. La librería MediaPipe detecta un total de veintiún puntos clave en cada mano, para un total de cuarenta y dos puntos clave por cada fotograma de video. Debido a que en LENSEGUA la posición de las manos también es relevante, se decidió calcular la posición relativa de cada punto clave con respecto a la cara, procesando las coordenadas en X y Y de cada punto clave. Cada uno de estos puntos cuenta con coordenadas en el eje x y en el eje y, lo cual resulta en un total de ochenta y cuatro características por fotograma de video.

5.5. Labeling de los datos

Después de procesar los videos y obtener las coordenadas de los puntos clave, se procedió a etiquetar los datos para que pudieran ser utilizados en el entrenamiento del modelo. Esto consistió en asignar una etiqueta a cada fotograma de video, indicando la seña correspondiente a la palabra que se estaba realizando en ese fotograma. Para esto, se utilizó la función `to_categorical` de la librería Keras de Python, la cual convierte un vector de etiquetas en una matriz binaria. Esta matriz binaria se utilizó para entrenar el modelo, permitiendo que este pudiera identificar las señas de la lengua de señas de Guatemala. Es importante mencionar que, para que las predicciones del modelo se puedan interpretar de forma correcta, se creó un diccionario que relaciona cada etiqueta con la palabra correspondiente.

5.6. Normalización de los datos

Para permitir que el modelo pudiera identificar las señas de forma en tiempo real, los datos se procesaron de forma que cada fotograma de video se convirtiera en una secuencia de datos que representara el movimiento de las manos. Tomando esto en cuenta, el conjunto de datos resultante consistió en una serie de secuencias de datos, cada una representando un fotograma de video y las coordenadas de los puntos clave detectados en ese fotograma. Después de calcular estas características, se procedió a normalizarlas para que todas las características tuvieran un valor entre menos uno y uno. Debido a que las coordenadas de los puntos clave podían ser negativas, para indicar que se encontraban a la izquierda o arriba de la cara, se utilizaron dos fórmulas diferentes para normalizar las coordenadas positivas y negativas.

Ecuación para normalizar coordenadas positivas:

$$x_{norm} = \frac{x}{|x_{min}|} \quad (5.1)$$

Ecuación para normalizar coordenadas negativas:

$$x_{norm} = \frac{x}{|x_{max}|} \quad (5.2)$$

Donde x_{norm} es el valor normalizado de la coordenada x , x es el valor de la coordenada x original, x_{min} es el valor mínimo de la coordenada en el conjunto de datos y x_{max} es el valor máximo de la coordenada en el conjunto de datos. Estas fórmulas se utilizaron para normalizar las coordenadas de los puntos clave en el eje x y en el eje y, lo cual permitió que todas las características tuvieran un valor entre -1 y 1. El resultado de este proceso fue un conjunto de datos con las características normalizadas, las cuales se utilizaron para entrenar el modelo. A continuación, se muestra un ejemplo de cómo se veían los datos después de ser procesados.

LABEL	COORDS
0	[0, -0.14792899408284024, 0.3733333333333335, -0...]
1	[0, -0.14792899408284024, 0.3733333333333335, -0...]
2	[0, -0.14595660749506903, 0.3724444444444447, -0...]
3	[0, -0.14595660749506903, 0.3715555555555553, -0...]
4	[0, -0.14595660749506903, 0.3706666666666664, -0...]

Figura 5.5: Ejemplo de datos procesados.

5.7. Oversampling

Debido al tamaño limitado del conjunto de datos, se utilizó la técnica de oversampling para aumentar la cantidad de datos disponibles para el entrenamiento del modelo. Esta técnica consistió en duplicar los datos existentes, lo cual permitió que el modelo pudiera entrenarse con una mayor cantidad de datos y mejorar su capacidad para identificar las señas de la lengua de señas de Guatemala. El oversampling se realizó copiando los datos existentes y agregando ruido a las coordenadas de los puntos clave, lo cual permitió que el modelo pudiera generalizar mejor a datos no vistos previamente. Con cada uno de los modelos entrenados, se evaluó el desempeño del modelo utilizando el conjunto de datos original y el conjunto de datos aumentado, lo cual permitió determinar si el oversampling había mejorado la capacidad del modelo para identificar las señas.

5.8. Entrenamiento del modelo

Para entrenar el modelo, se utilizó el 80 % del conjunto de datos procesado en la fase anterior, mientras que el 20 % restante se utilizó para validar el modelo. Durante este proceso, se probaron diferentes hiperparámetros y técnicas de entrenamiento para encontrar la combinación que ofreciera el mejor desempeño. Adicionalmente, una vez seleccionada una red neuronal, se evaluó el desempeño de esta de forma continua para poder hacer ajustes y optimizaciones que mejoraran sus resultados.

Debido a la naturaleza de los datos, se decidió utilizar una red neuronal feedforward para entrenar el modelo. Este modelo se utilizó para entrenar un clasificador que pudiera identificar las señas de la

lengua de señas de Guatemala utilizando los datos procesados en la fase anterior. En cada iteración, se ajustaron los hiperparámetros de la red neuronal, como el número de capas, el número de neuronas por capa y la tasa de aprendizaje, con el objetivo de mejorar la precisión del modelo. Adicionalmente, se probaron diferentes funciones de activación y optimizadores para encontrar la combinación que resultara en el mejor desempeño.

```
model = tf.keras.Sequential()
model.add(layers.Input(shape=(data.shape[1],)))
model.add(layers.Dense(128, activation='relu'))
model.add(layers.Dense(64, activation='relu'))
model.add(layers.Dense(numClasses, activation='softmax'))
```

Figura 5.6: Ejemplo de una red neuronal feedforward sencilla.

Con el objetivo de mejorar la capacidad del modelo para generalizar a datos no vistos previamente, se utilizó la técnica de dropout durante el entrenamiento. Esta técnica consistió en desactivar aleatoriamente un porcentaje de las neuronas en cada iteración, lo cual permitió que el modelo no dependiera de una sola característica para hacer predicciones. Adicionalmente, se probaron diferentes funciones de activación y optimizadores para encontrar la combinación que resultara en el mejor desempeño. Debido a la naturaleza de los datos, se decidió utilizar la función de activación tangente hiperbólica, ya que esta función es adecuada para datos normalizados entre -1 y 1.

```
model = tf.keras.Sequential()
model.add(layers.Input(shape=(data.shape[1],)))
model.add(layers.Dense(512, activation='tanh'))
model.add(layers.Dropout(0.2))
model.add(layers.Dense(256, activation='tanh'))
model.add(layers.Dropout(0.2))
model.add(layers.Dense(128, activation='tanh'))
model.add(layers.Dropout(0.1))
model.add(layers.Dense(64, activation='tanh'))
model.add(layers.Dropout(0.1))
model.add(layers.Dense(numClasses, activation='softmax'))
```

Figura 5.7: Implementación de mejoras en la red neuronal feedforward.

5.9. Evaluación del modelo

Como último paso, se evaluó la habilidad del modelo para reconocer diferentes señas de la lengua de señas de Guatemala. Para este proceso, se utilizó una serie de videos que no se habían utilizado en el entrenamiento del modelo, los cuales se procesaron de la misma forma que los videos de entrenamiento. La primera de estas evaluaciones se realizó con videos que contenían una sola seña, lo cual permitió evaluar la capacidad del modelo para identificar señas individuales. Posteriormente, se evaluó el modelo con videos que contenían frases completas, lo cual permitió evaluar la capacidad del modelo para identificar frases completas.

Como métrica de evaluación, se calculó el porcentaje de videos en los cuales el modelo identificó de manera correcta la palabra correspondiente a la seña. Adicionalmente, se buscaron errores comunes cometidos por el modelo para que estos pudieran ser corregidos. Un ejemplo de esto sería si el modelo constantemente confundía una seña por otra, lo cual indicaría que el modelo necesitaba más

datos correspondientes a una de esas señas o que se debía modificar el modelo. En este proceso de validación manual, resultó útil la codificación con la cual se identificaron los videos en la fase de preparación del conjunto de datos.

$$\text{Precisión} = \frac{\text{Número de predicciones correctas}}{\text{Número total de predicciones}} \quad (5.3)$$

Adicional a la precisión, se obtuvo la matriz de confusión, la cual permitió identificar si existían señas que eran constantemente confundidas por el modelo. Utilizando esta información, se ajustaron los hiperparámetros del modelo y se realizaron mejoras en el conjunto de datos para corregir los errores comunes cometidos por el modelo. La matriz de confusión también permitió calcular la sensibilidad y el puntaje F1, los cuales se utilizaron como métricas adicionales para evaluar el desempeño del modelo.

Debido a que el tiempo de procesamiento del modelo era crucial para el éxito del proyecto, se calculó la velocidad de procesamiento del modelo, en fotogramas por segundo. Los videos que se utilizaron en este proyecto fueron grabados a 30 fotogramas por segundo, por lo que se esperaba que el modelo pudiera procesar los videos a una velocidad similar. Esta métrica, de fotogramas procesados por segundo, se calculó utilizando la siguiente fórmula:

$$\text{Velocidad de procesamiento} = \frac{\text{Número de fotogramas procesados}}{\text{Tiempo de procesamiento}} \quad (5.4)$$

5.10. Evaluación del conjunto de datos

Para evaluar el conjunto de datos, se utilizó un analizador de componentes principales (PCA) para reducir la dimensionalidad de los datos y visualizarlos en un espacio de dos dimensiones. El PCA permitió identificar si los datos se agrupaban de forma coherente, lo cual indicaría que el modelo tendría una alta precisión al identificar las señas. Debido a que el PCA se realizó con los datos de todos los fotogramas de video, se esperaba que los datos estuvieran algo dispersos, ya que cada fotograma de video representaba un punto en el espacio de dos dimensiones. Lo que se buscaba con el PCA era identificar si existían señas que se agrupaban de forma similar, lo cual indicaría que el modelo tendría problemas para distinguir entre esas señas.

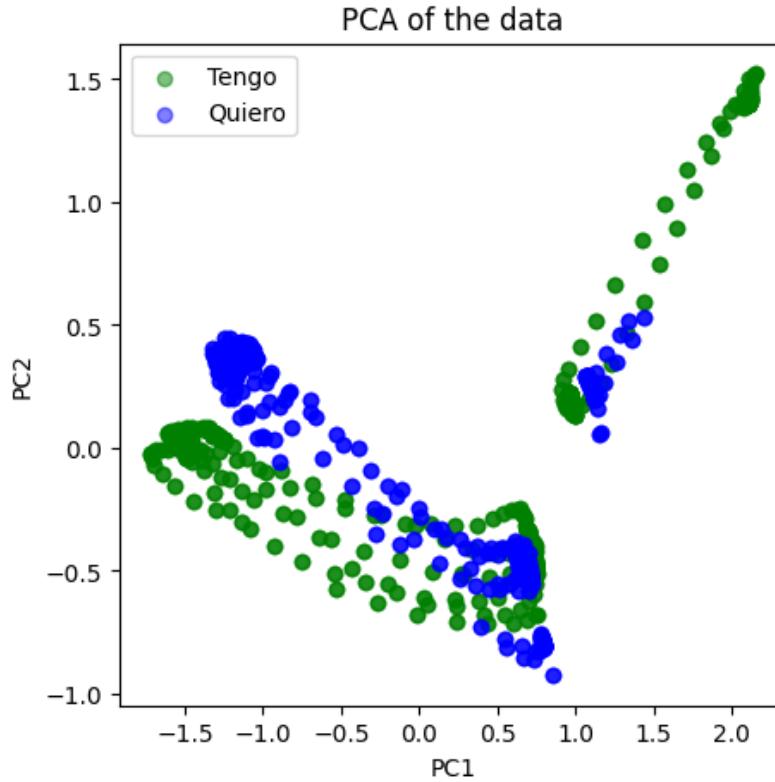


Figura 5.8: Ejemplo de visualización de datos con PCA.

Durante el proceso de entrenamiento del modelo, se identificaron algunas señas que eran comúnmente confundidas por el modelo. En estos casos, el análisis de los datos con PCA permitió identificar si existían diferencias significativas entre las señas, lo cual permitió identificar si el problema era causado por la similitud entre las señas o por las fallas en el modelo. En la figura 5.8 se muestra un ejemplo de cómo se identificaron señas que no se distinguían claramente en el espacio de dos dimensiones.

Utilizando la matriz de confusión y el análisis de los datos con PCA, se buscaba identificar si los errores cometidos por el modelo eran causados por la similitud entre las señas o por fallas en el modelo. En el caso de que los errores fueran causados por la similitud entre las señas, se esperaba que tanto la matriz de confusión como el análisis de los datos con PCA mostraran que las señas se agrupaban de forma similar. En el caso de que los errores fueran causados por fallas en el modelo, se esperaba que la matriz de confusión mostrara que el modelo cometía errores aleatorios, mientras que el análisis de los datos con PCA mostrara que las señas se agrupaban de forma coherente. La figura 5.9 muestra un ejemplo de la matriz de confusión utilizada para evaluar el desempeño del modelo.

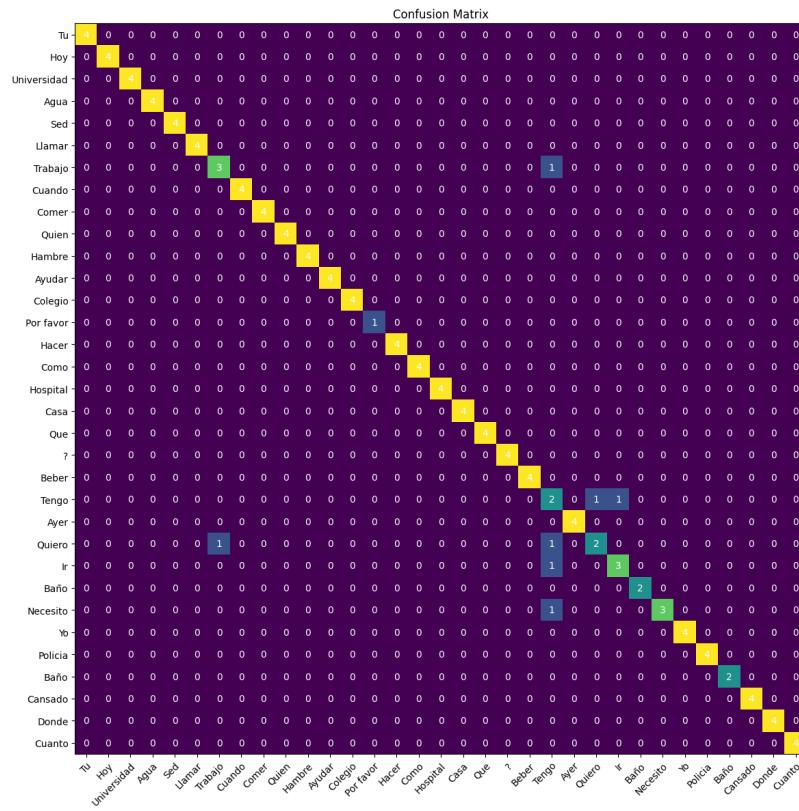


Figura 5.9: Ejemplo de matriz de confusión.

5.11. Publicación del modelo

Una vez que el modelo fue entrenado y evaluado, se procedió a publicarlo en un servidor web para que pudiera ser utilizado por los usuarios. Para esto, se utilizó la librería Flask de Python, la cual permitió crear una API que se comunicara con el modelo y que pudiera ser accedida por los usuarios a través de una interfaz web. La API permitió que los usuarios enviaran videos a través de la interfaz web, los cuales se procesaban con el modelo y se devolvía la palabra correspondiente a la señal identificada. Esto requirió de un programa adicional, diseñado para procesar los videos enviados por los usuarios y enviarlos a la API para su procesamiento. Este programa carga el modelo entrenado, al igual que algunos parámetros adicionales, y se encarga de recibir los videos de los usuarios, procesarlos con el modelo, y devolver la palabra correspondiente a la señal identificada. Para validar el funcionamiento de la Interfaz de Programación de Aplicaciones (API), se realizaron pruebas con videos de prueba utilizando Postman.

CAPÍTULO 6

Antecedentes

Puede encontrarse un trabajo similar en la tesis de José Eduardo López Gómez[18], en la cual se desarrolló un modelo de reconocimiento del abecedario de la lengua de señas de Guatemala. En este trabajo, se recopilaron datos de 29 letras del abecedario de la lengua de señas de Guatemala, las cuales fueron utilizadas para entrenar un modelo de reconocimiento de lengua de señas. Al igual que en este proyecto, se utilizó la librería MediaPipe para el reconocimiento de gestos, y se utilizó un modelo de aprendizaje profundo para el reconocimiento de las señas.

CAPÍTULO 7

Alcance

El proyecto "Señas Chapinas: Traductor de LENSEGUA" tiene como objetivo desarrollar un sistema de visión por computadora para el reconocimiento de la lengua de señas de Guatemala (LENSEGUA), facilitando la comunicación entre personas con discapacidad auditiva y el resto de la población. Para ello, se delimitará un universo de al menos veinticinco palabras de uso cotidiano, las cuales serán representadas a través de videos que se recopilarán y procesarán utilizando la librería MediaPipe de Google. Este sistema se entrenará utilizando el ochenta por ciento de los datos recopilados, mientras que el veinte por ciento restante se destinará a la validación del modelo, garantizando su precisión y eficiencia en el reconocimiento de señas en tiempo real.

El alcance del proyecto no solo incluye la creación de un modelo que reconozca las señas, sino también la implementación de este sistema en dispositivos móviles, promoviendo la inclusión social y el aprendizaje de la lengua de señas en Guatemala. Adicionalmente, se desarrollará un API que permita la integración de este sistema en aplicaciones de terceros, facilitando su uso en diferentes contextos y plataformas. A través de este enfoque, se busca contribuir a la eliminación de barreras comunicativas, mejorando la calidad de vida de las personas con discapacidad auditiva y fomentando una mayor equidad en la comunicación en el país.

CAPÍTULO 8

Resultados

8.1. Conjunto de datos

8.1.1. Datos de lengua de señas de Guatemala

La recopilación de datos era un objetivo importante para este proyecto, debido a que no existen conjuntos de datos públicos de lengua de señas de Guatemala. Esto representaba un desafío importante para el desarrollo de un modelo de reconocimiento de lengua de señas, ya que no se contaba con datos para entrenar un modelo de aprendizaje. Siguiendo la metodología propuesta para el proyecto, se recopilaron datos de 32 palabras de la lengua de señas de Guatemala, las cuales son: *agua, ayer, ayudar, baño, beber, cansado, casa, colegio, comer, como, cuando, cuanto, donde, hacer, hambre, hospital, hoy, ir, llamar, necesito, policía, por favor, que, quien, quiero, sed, tengo, trabajo, tu, universidad, yo, y el signo de interrogación.*

El conjunto de datos está compuesto de 30 videos por palabra, los cuales fueron editados manualmente para incluir solo el gesto de la palabra correspondiente. En total, se recopilaron 960 videos, los cuales fueron divididos en dos conjuntos: uno de entrenamiento y otro de prueba. Utilizando una proporción de 80 % para el conjunto de entrenamiento y 20 % para el conjunto de prueba. Con esta proporción, el conjunto de entrenamiento está compuesto de 768 videos, mientras que el conjunto de prueba está compuesto de 192 videos. Con el objetivo de permitir que otros investigadores puedan utilizar el conjunto de datos, se ha publicado en el siguiente enlace: <https://github.com/Ldsc2002/lensegua-dataset>.

8.1.2. Análisis del conjunto de datos

En base a la matriz de confusión del modelo final, se puede observar que el modelo tiene un buen desempeño en la mayoría de las clases. Sin embargo, se puede observar que el modelo ocasionalmente confunde algunas clases, como *sed* y *quien*. Esto puede deberse a que los gestos de estas palabras son similares, lo cual dificulta la distinción entre ellas. Para validar esta hipótesis, se realizó un análisis de similitud entre las clases, utilizando un análisis de componentes principales (PCA).

Se identificó que el modelo confunde las siguientes clases: *como y agua*, *como y beber*, *cuando y donde*, y *sed y quien*. A continuación se presentan los resultados del análisis de similitud entre las clases para cada par de clases.

En la figura 8.1 se puede observar que las clases *como y agua* tienen una alta similitud en los movimientos de una mano. Aunque la otra mano tiene un movimiento distinto en cada clase, el alto grado de similitud en una mano puede dificultar la distinción entre ellas.

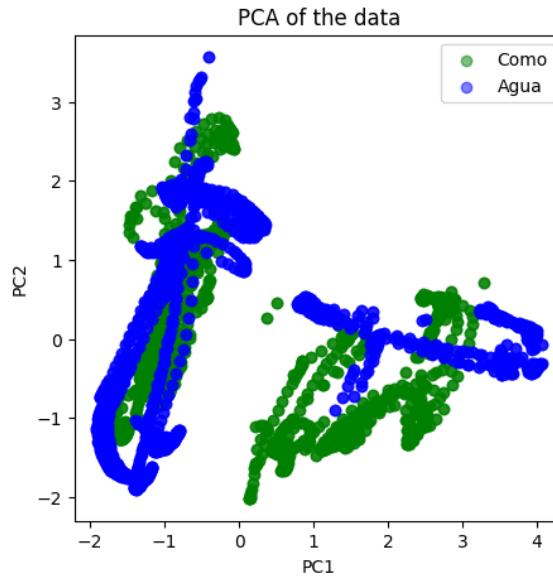


Figura 8.1: Análisis de similitud entre las clases *como y agua*

En la figura 8.2 se puede observar que las clases *como y beber* tienen cierto grado de similitud en los movimientos de ambas manos. Esto puede dificultar la distinción entre ellas, ya que el modelo puede confundir los movimientos de ambas manos.

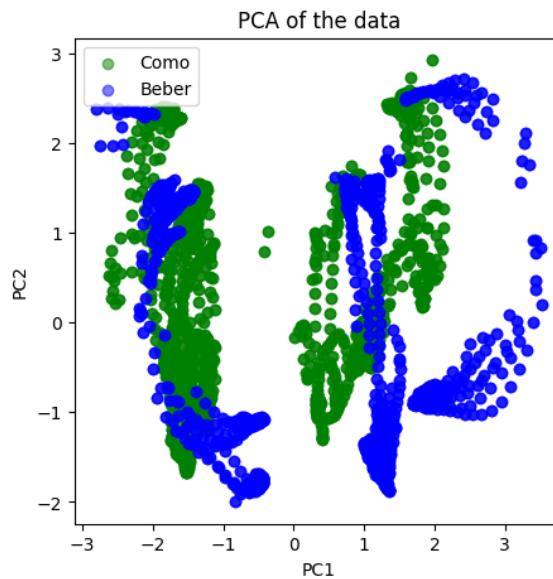


Figura 8.2: Análisis de similitud entre las clases *como y beber*

En la figura 8.3 se puede observar que las clases *cuando* y *donde* tienen un bajo grado de similitud en los movimientos de ambas manos. Sin embargo, en el análisis de similitud se observa que las clases tienen cierto grado de superposición, lo cual puede dificultar la distinción entre ellas.

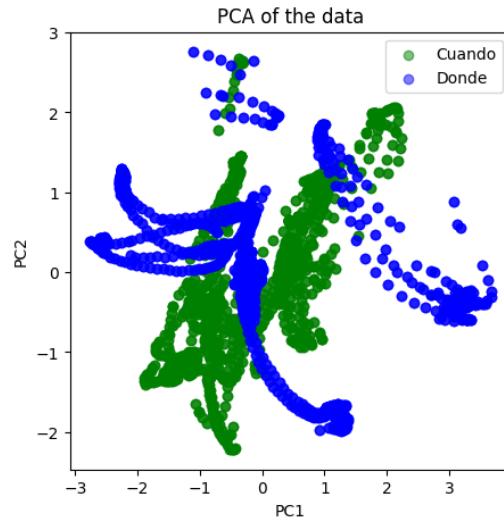


Figura 8.3: Análisis de similitud entre las clases *cuando* y *donde*

En la figura 8.4 se puede observar que las clases *sed* y *quien* tienen un bajo grado de similitud en los movimientos de ambas manos. El análisis de similitud no muestra una superposición significativa entre las clases, lo cual indica que el modelo puede confundir las clases debido a otros factores.

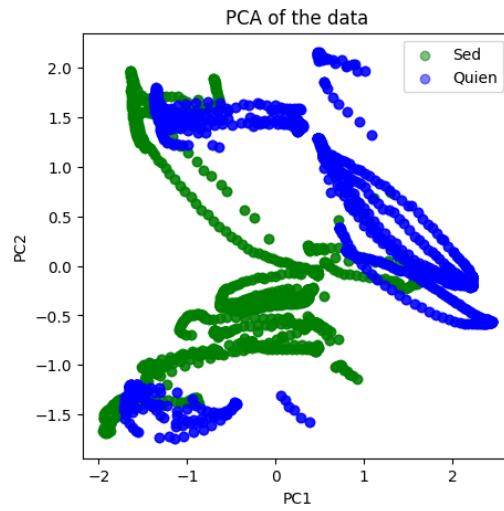


Figura 8.4: Análisis de similitud entre las clases *sed* y *quien*

Adicional a esto, se realizó un análisis de similitud entre todas las clases, con el objetivo de identificar otras clases que puedan tener un alto grado de similitud. En la figura 8.5 se puede observar el análisis de similitud entre todas las clases.

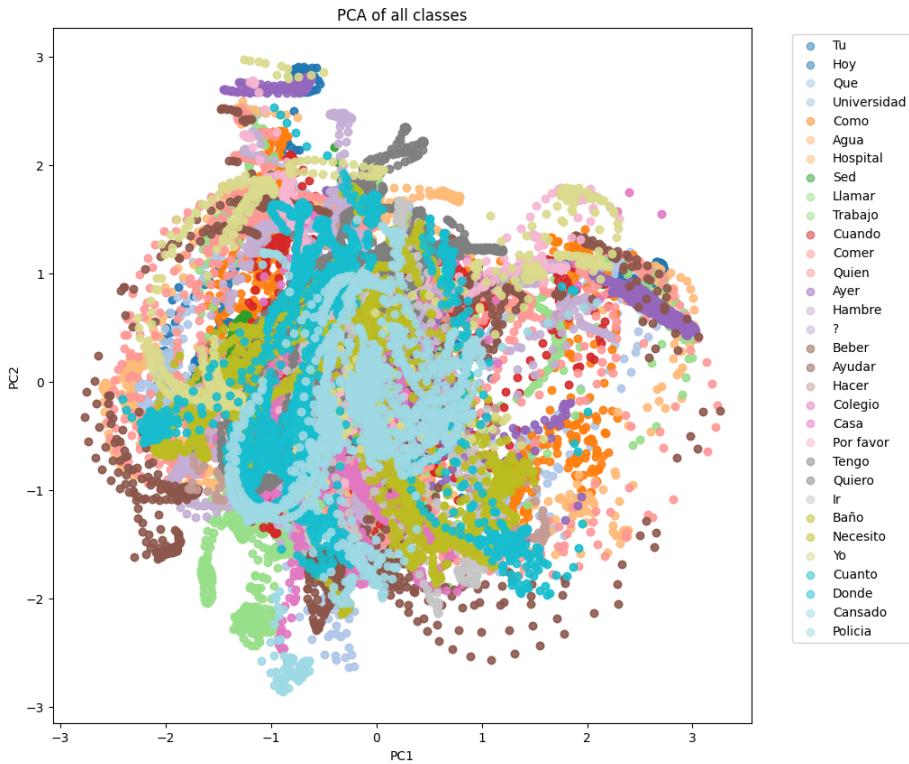


Figura 8.5: Análisis de similitud entre todas las clases

Por último, se validó el balance de las clases en el conjunto de datos de entrenamiento. La figura 8.6 muestra el balance de las clases en el conjunto de datos de entrenamiento.

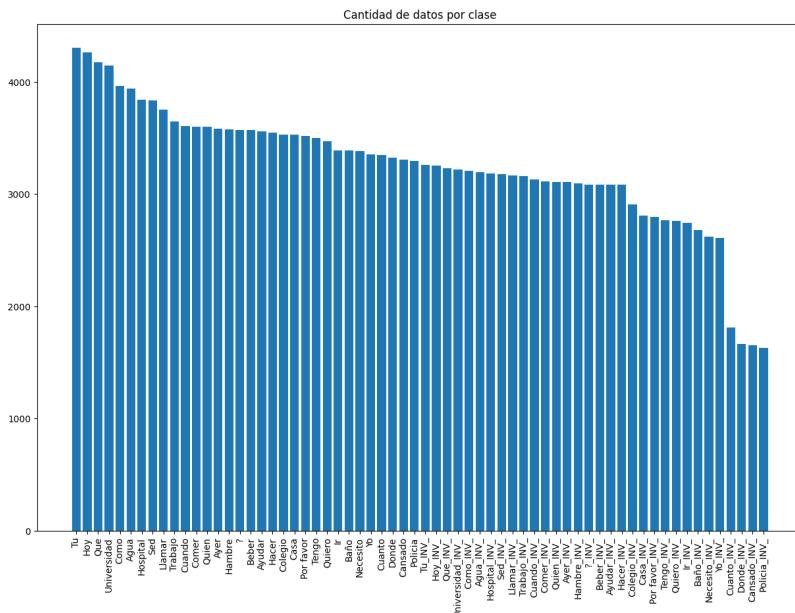


Figura 8.6: Balance de clases en el conjunto de datos de entrenamiento

8.2. Proceso iterativo de desarrollo del modelo

El proceso de desarrollo del modelo de reconocimiento de lengua de señas de Guatemala fue iterativo, siguiendo la metodología propuesta para el proyecto. En total, se realizaron 8 iteraciones, en las cuales se crearon modelos distintos y se evaluaron utilizando el conjunto de prueba. Todos los modelos fueron entrenados con 25 épocas, utilizando un tamaño de lote de 32 y un optimizador Adam. A continuación se presentan los modelos resultantes de cada iteración, junto con su desempeño en el conjunto de prueba.

8.2.1. Modelo base

La composición del modelo base se puede observar en la figura 8.7. Este modelo tiene una capa de entrada, cuatro capas densas con activación tangente hiperbólica y una capa de salida con activación softmax.

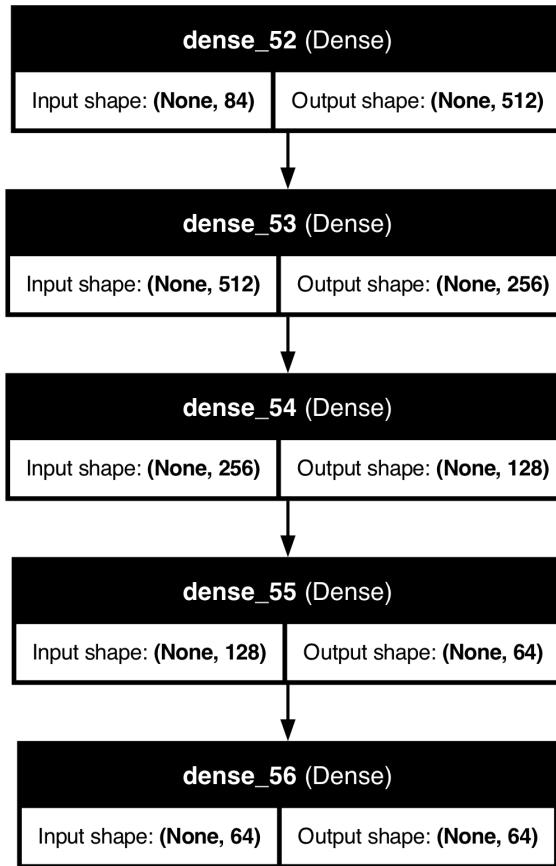


Figura 8.7: Modelo inicial

El desempeño del modelo inicial se puede observar en la tabla 8.1. Este modelo inicial tiene una precisión de 0.7708, una sensibilidad de 0.8822 y un F1 de 0.8895. Adicional a esto, se puede observar la matriz de confusión en la figura 8.9 y el historial de entrenamiento en la figura 8.8.

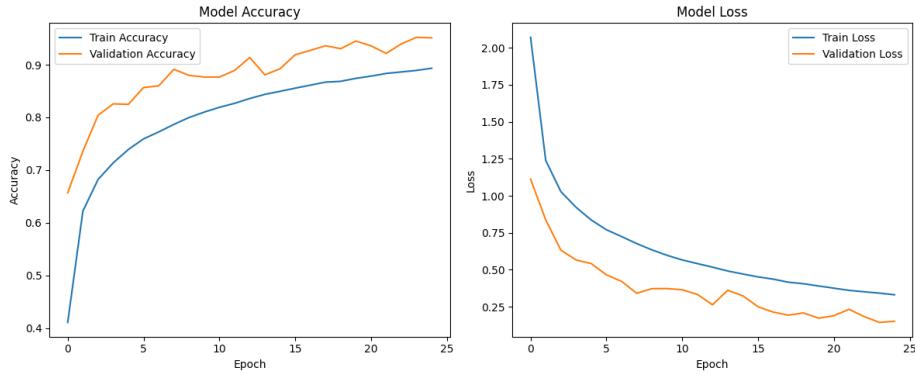


Figura 8.8: Historial de entrenamiento del modelo base

Métrica	Valor
Precisión	0.7708
Sensibilidad	0.8822
F1	0.8895

Tabla 8.1: Desempeño del modelo base

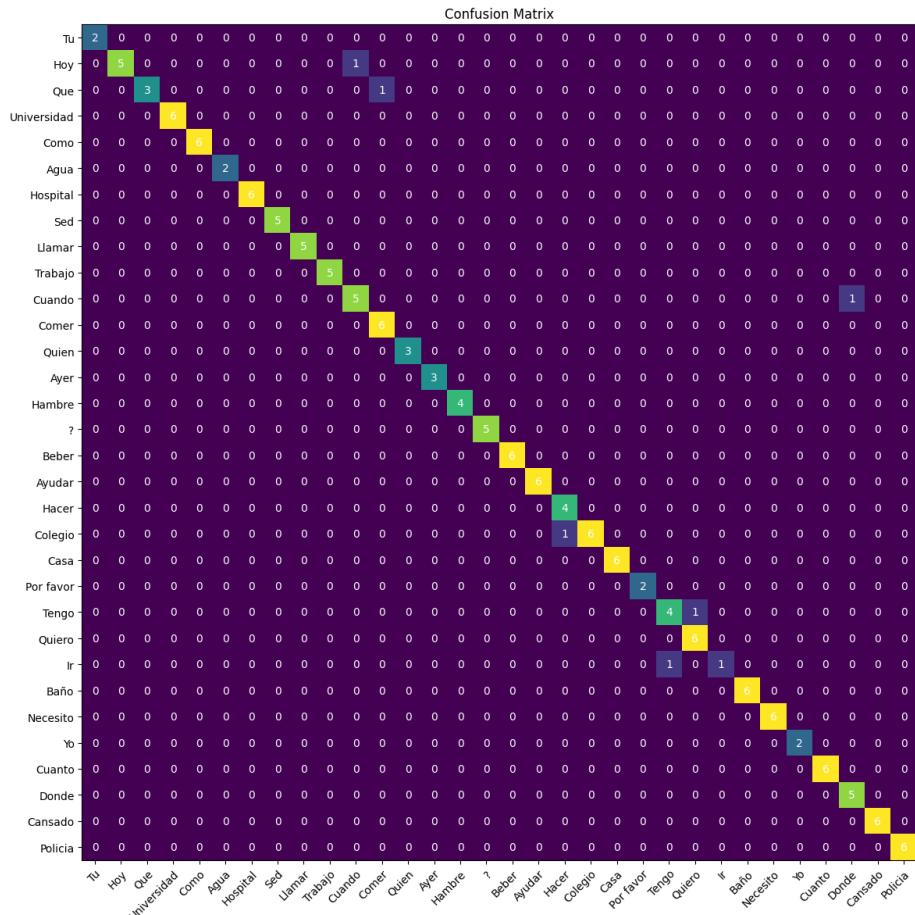


Figura 8.9: Matriz de confusión del modelo base

8.2.2. Aumento de complejidad del modelo base

En la iteración previa, se observó que el modelo base tenía un desempeño aceptable, pero se consideró que la arquitectura del modelo era muy simple. En esta iteración, se aumentó la complejidad del modelo base, añadiendo dos capas densas adicionales con activación tangente hiperbólica. Estas capas adicionales tienen 2048 y 1024 neuronas, respectivamente.

La composición del modelo con aumento de complejidad se puede observar en la figura 8.10. Este modelo tiene una capa de entrada, seis capas densas con activación tangente hiperbólica y una capa de salida con activación softmax.

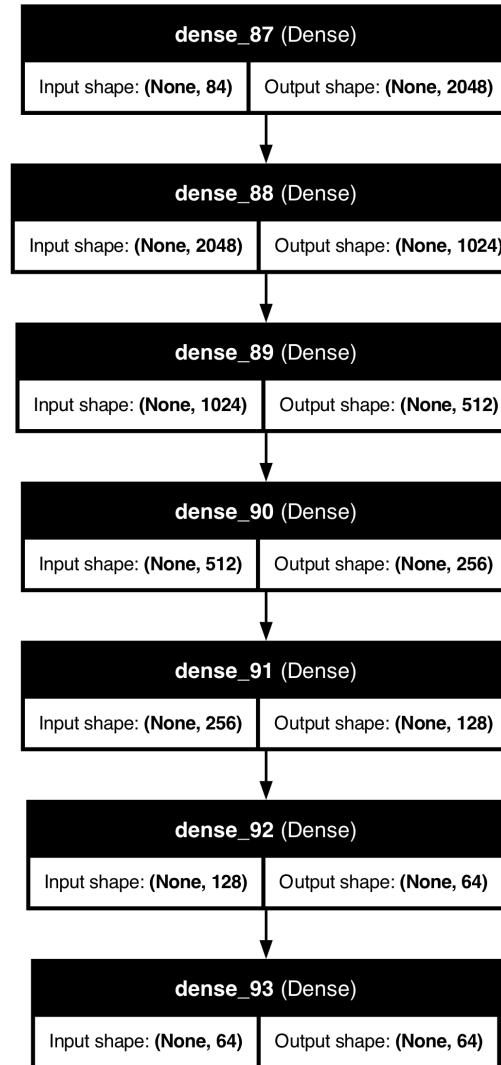


Figura 8.10: Modelo con aumento de complejidad

El desempeño de este modelo se puede observar en la tabla 8.2. Este modelo tiene una precisión de 0.5937, una sensibilidad de 0.8957 y un F1 de 0.9026. Adicional a esto, se puede observar la matriz de confusión en la figura 8.12 y el historial de entrenamiento en la figura 8.11.

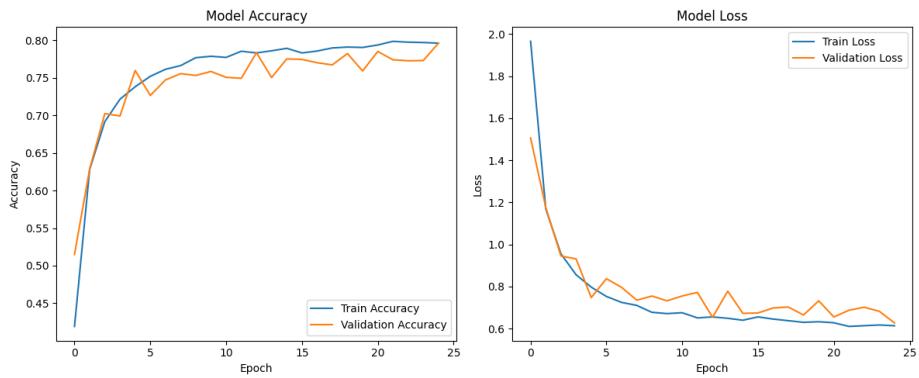


Figura 8.11: Historial de entrenamiento del modelo con aumento de complejidad

Métrica	Valor
Precisión	0.5937
Sensibilidad	0.8957
F1	0.9026

Tabla 8.2: Desempeño del modelo con aumento de complejidad

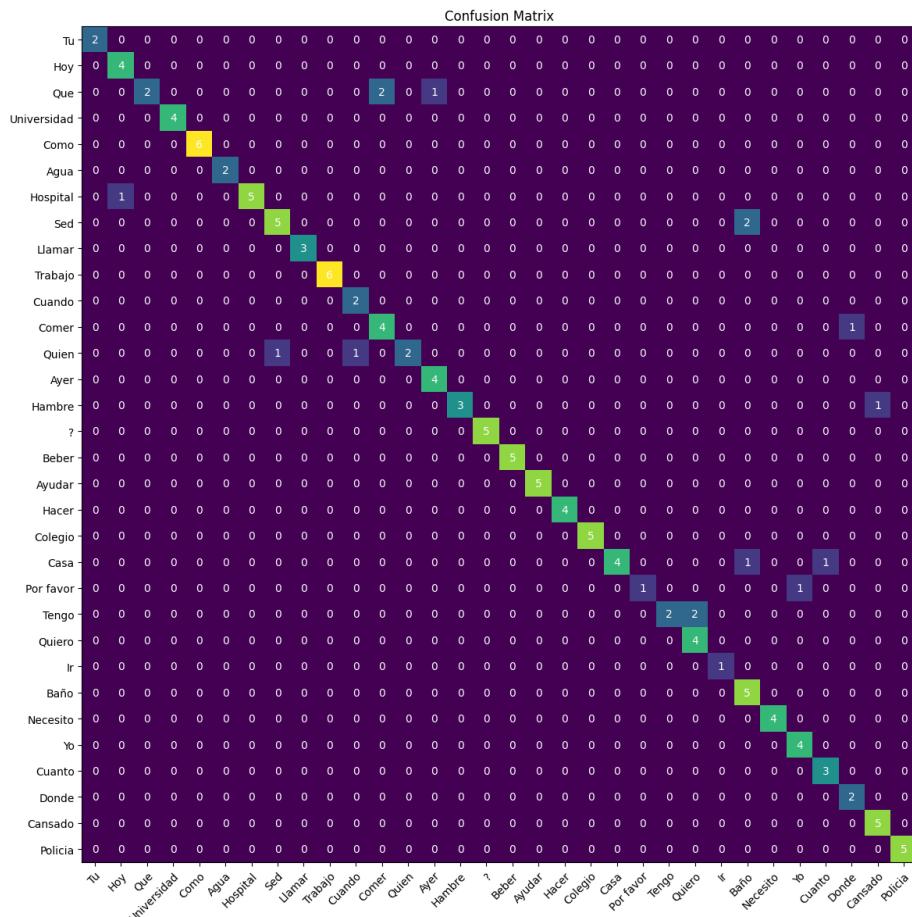


Figura 8.12: Matriz de confusión del modelo con aumento de complejidad

8.2.3. Adición de dropout al modelo base

La composición del modelo base con dropout se puede observar en la figura 8.13. Este modelo tiene una capa de entrada, cuatro capas densas con activación tangente hiperbólica y una capa de salida con activación softmax. Adicional a esto, se ha añadido una capa de dropout con una tasa de 0.1 a cada capa densa.

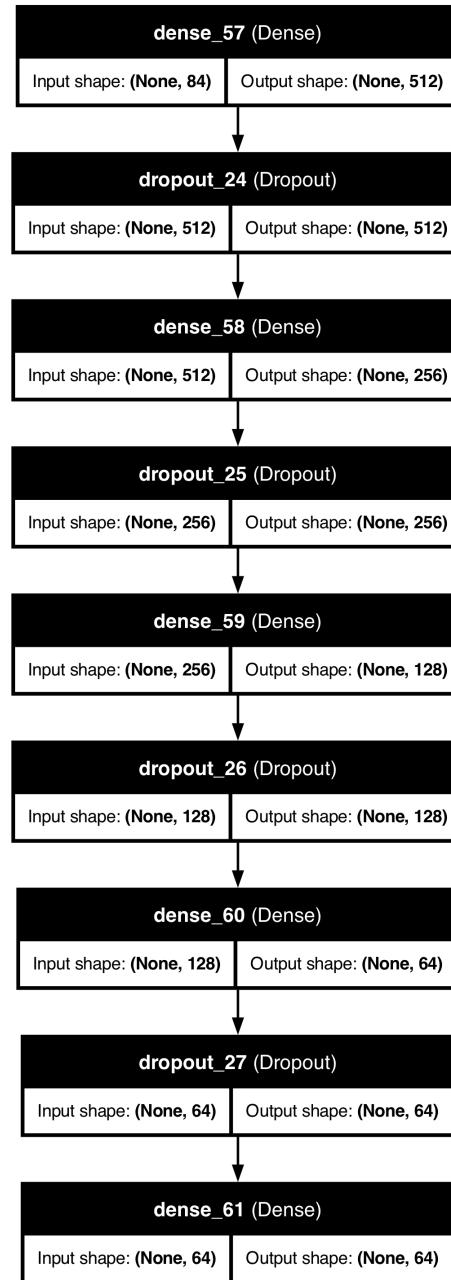


Figura 8.13: Modelo base con dropout

El desempeño de este modelo se puede observar en la tabla 8.3. Este modelo tiene una precisión de 0.8072, una sensibilidad de 0.9247 y un F1 de 0.9274. Adicional a esto, se puede observar la matriz de confusión en la figura 8.15 y el historial de entrenamiento en la figura 8.14.

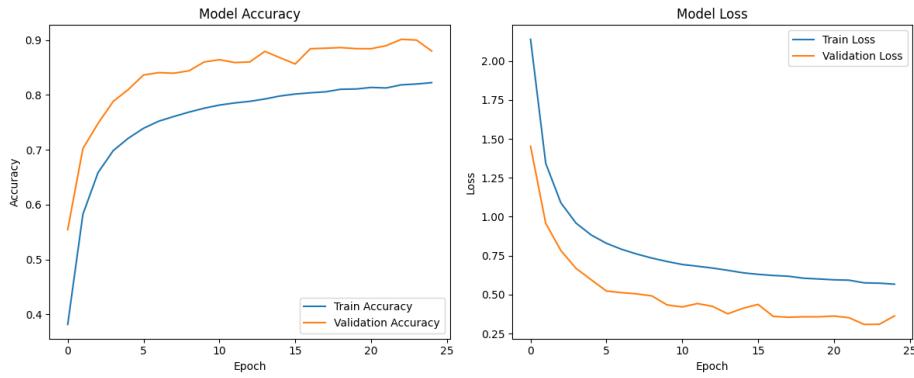


Figura 8.14: Historial de entrenamiento del modelo base con dropout

Métrica	Valor
Precisión	0.8072
Sensibilidad	0.9247
F1	0.9274

Tabla 8.3: Desempeño del modelo base con dropout

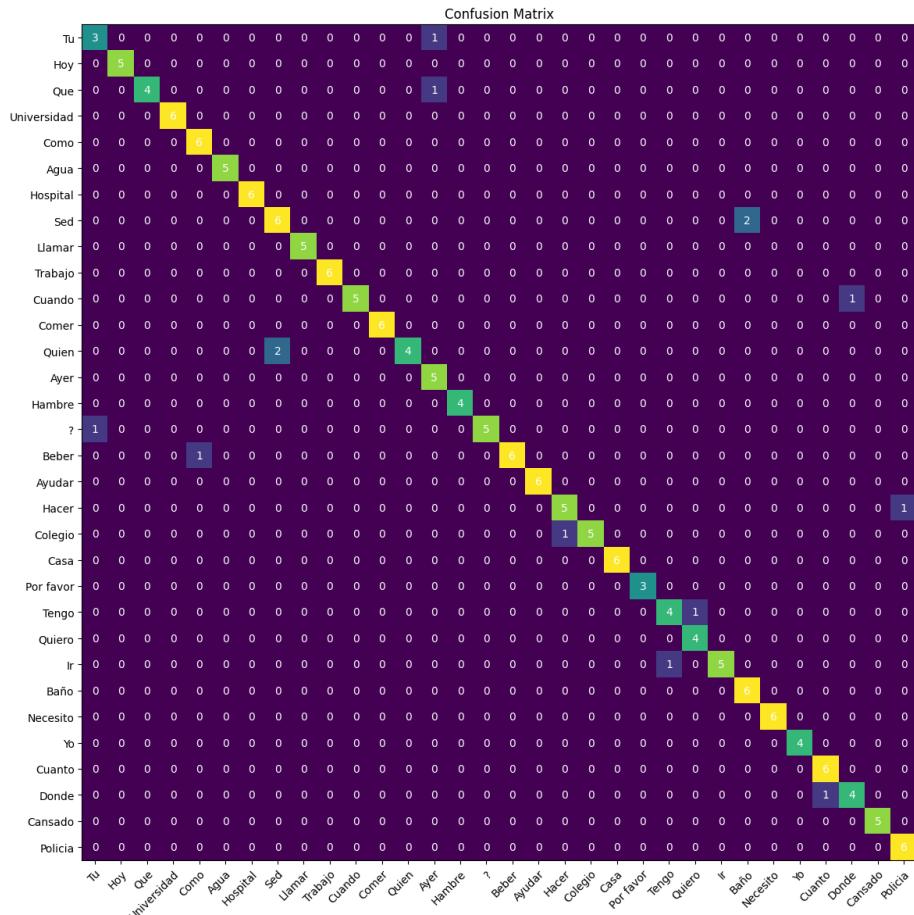


Figura 8.15: Matriz de confusión del modelo base con dropout

8.2.4. Fine tuning de las capas dropout

En la iteración previa, se observó que la adición de dropout al modelo base mejoró el desempeño del modelo. Esto se debe a que el dropout ayuda a prevenir el sobreajuste del modelo, al desactivar aleatoriamente un porcentaje de las neuronas en cada época, lo cual permite que el modelo generalice mejor. En esta iteración, se realizó un fine tuning de la tasa de dropout, con el objetivo de encontrar la tasa de dropout que maximizara el desempeño del modelo. Se realizan pruebas con dos modelos distintos, con el objetivo de encontrar la tasa de dropout que maximice el desempeño del modelo.

La composición del primero de los modelos se puede observar en la figura 8.16. Este modelo tiene una capa de entrada, cuatro capas densas con activación tangente hiperbólica y una capa de salida con activación softmax. A diferencia del modelo anterior, este modelo tiene una capa de dropout con una tasa de 0.2 a cada capa densa.

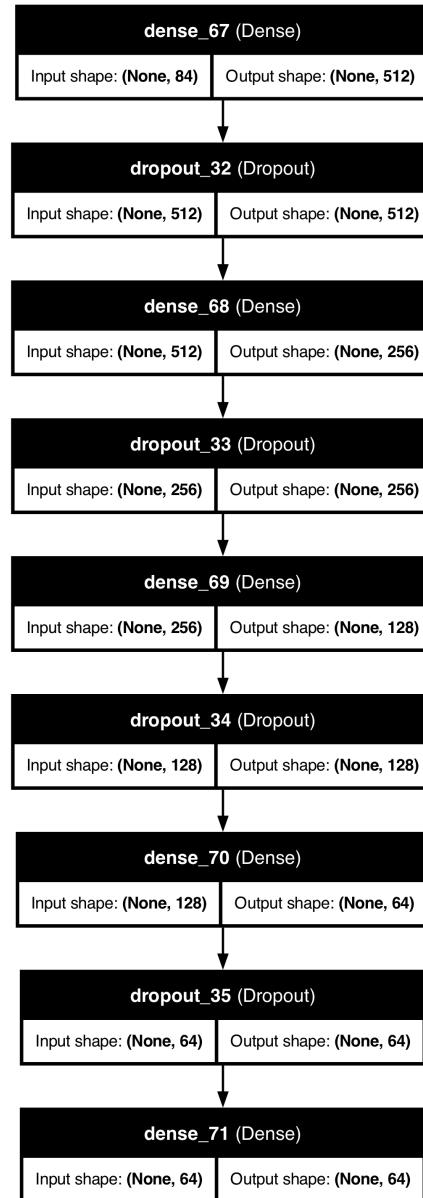


Figura 8.16: Primera iteración de fine tuning de la tasa de dropout

El desempeño de este modelo se puede observar en la tabla 8.4. Este modelo tiene una precisión de 0.7656, una sensibilidad de 0.9681 y un F1 de 0.9617. Adicional a esto, se puede observar la matriz de confusión en la figura 8.18 y el historial de entrenamiento en la figura 8.17.

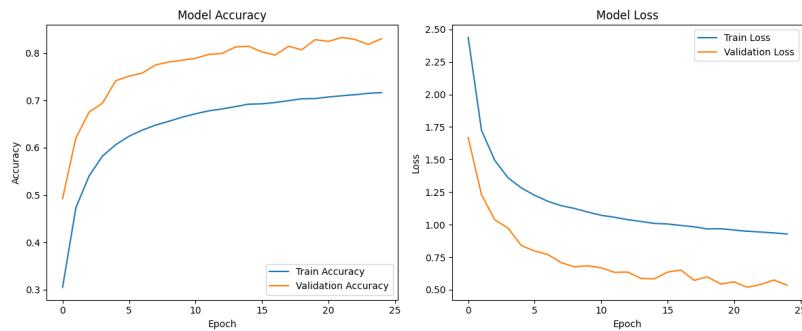


Figura 8.17: Historial de entrenamiento del primer modelo de fine tuning de la tasa de dropout

Métrica	Valor
Precisión	0.7656
Sensibilidad	0.9681
F1	0.9617

Tabla 8.4: Desempeño del primer modelo de fine tuning de la tasa de dropout

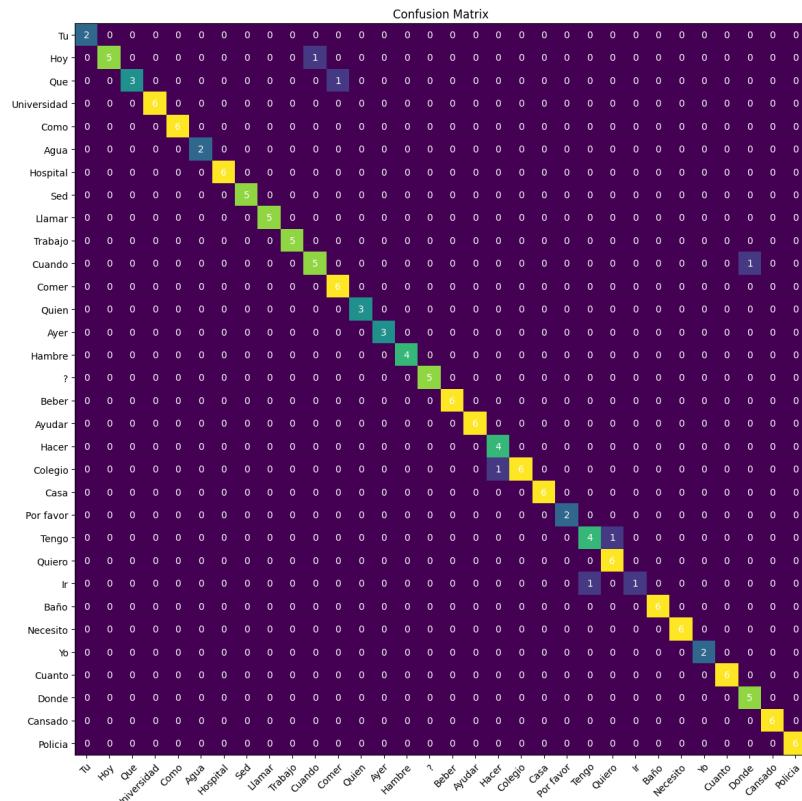


Figura 8.18: Matriz de confusión del primer modelo de fine tuning de la tasa de dropout

La composición del segundo de los modelos se puede observar en la figura 8.19. Este modelo tiene una capa de entrada, cuatro capas densas con activación tangente hiperbólica y una capa de salida con activación softmax. Debido a que el modelo anterior tuvo un desempeño inferior al modelo base con dropout, se decidió probar con dos capas de dropout con una tasa de 0.2 y dos capas de dropout con una tasa de 0.1.

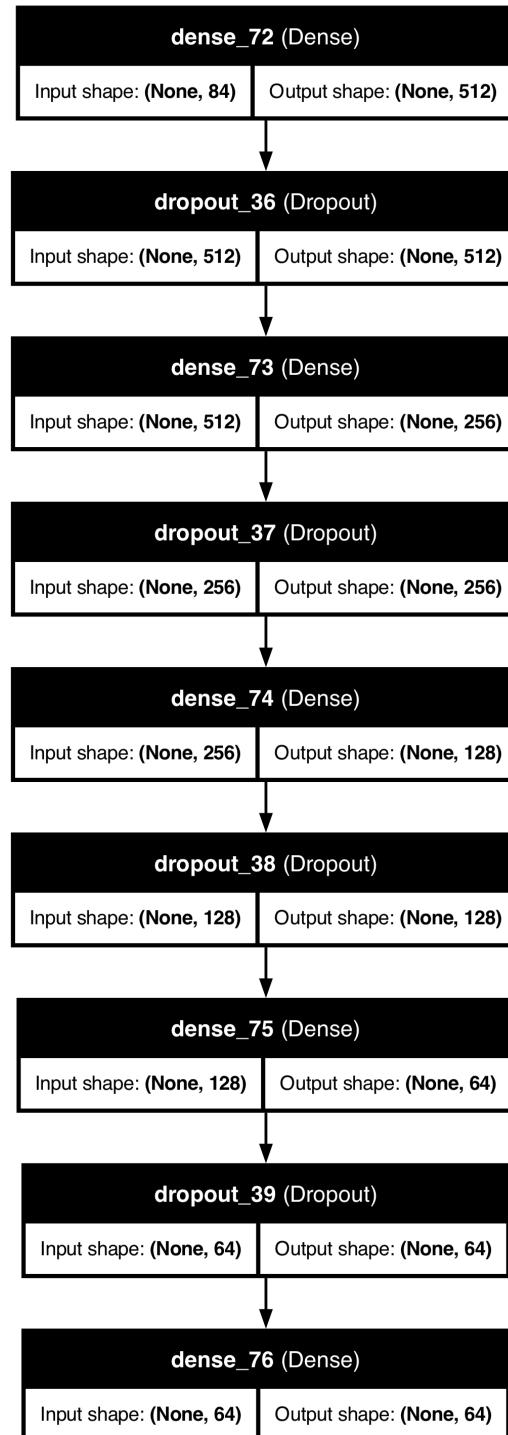


Figura 8.19: Segunda iteración de fine tuning de la tasa de dropout

El desempeño de este modelo se puede observar en la tabla 8.5. Este modelo tiene una precisión de 0.7708, una sensibilidad de 0.9541 y un F1 de 0.9543. Adicional a esto, se puede observar la matriz de confusión en la figura 8.21 y el historial de entrenamiento en la figura 8.20.

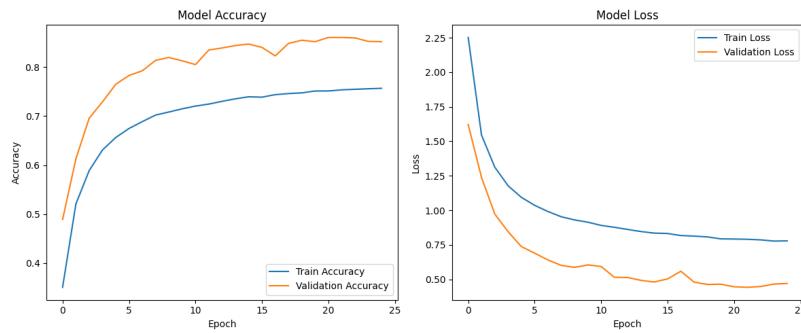


Figura 8.20: Historial de entrenamiento del segundo modelo de fine tuning de la tasa de dropout

Métrica	Valor
Precisión	0.7708
Sensibilidad	0.9541
F1	0.9543

Tabla 8.5: Desempeño del segundo modelo de fine tuning de la tasa de dropout

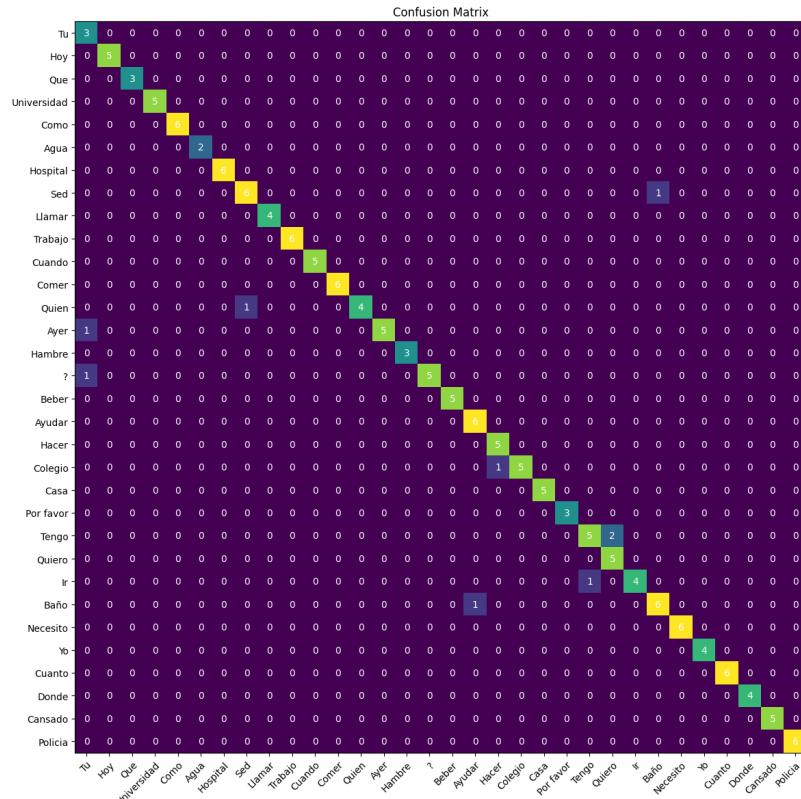


Figura 8.21: Matriz de confusión del segundo modelo de fine tuning de la tasa de dropout

8.2.5. Adición de normalización por lotes al modelo base

La normalización por lotes es una técnica que se utiliza para normalizar las entradas de una red neuronal, con el objetivo de acelerar el proceso de entrenamiento y mejorar la precisión del modelo. En esta iteración, se añadió una capa de normalización por lotes por cada capa densa del modelo base, con el objetivo de evaluar si la normalización por lotes mejoraba el desempeño del modelo. La composición del modelo con normalización por lotes se puede observar en la figura 8.22.

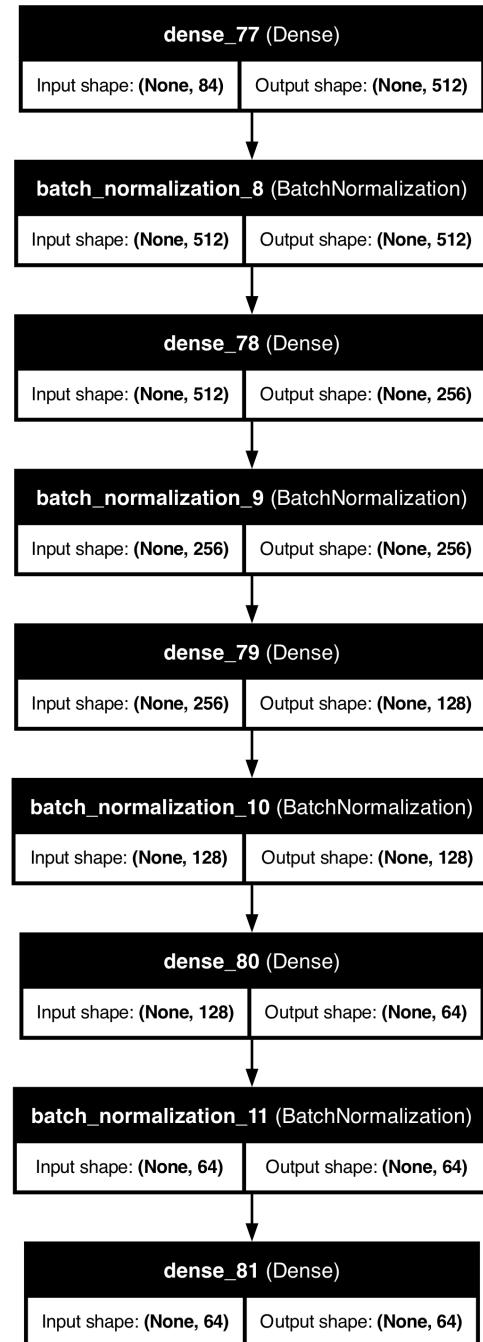


Figura 8.22: Modelo con normalización por lotes

El desempeño de este modelo se puede observar en la tabla 8.6. Este modelo tiene una precisión de 0.8593, una sensibilidad de 0.9136 y un F1 de 0.9155. Adicional a esto, se puede observar la matriz de confusión en la figura 8.24 y el historial de entrenamiento en la figura 8.23.

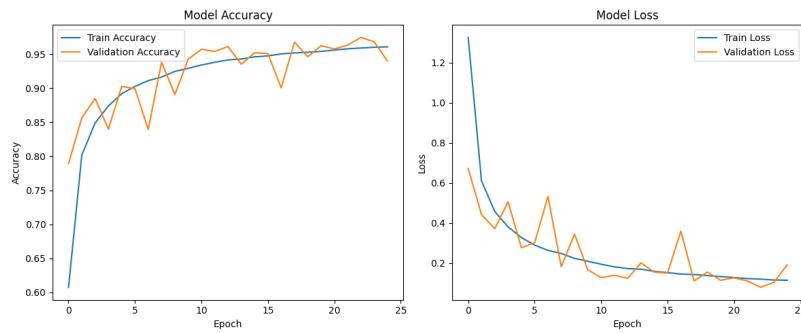


Figura 8.23: Historial de entrenamiento del modelo con normalización por lotes

Métrica	Valor
Precisión	0.8593
Sensibilidad	0.9136
F1	0.9155

Tabla 8.6: Desempeño del modelo con normalización por lotes

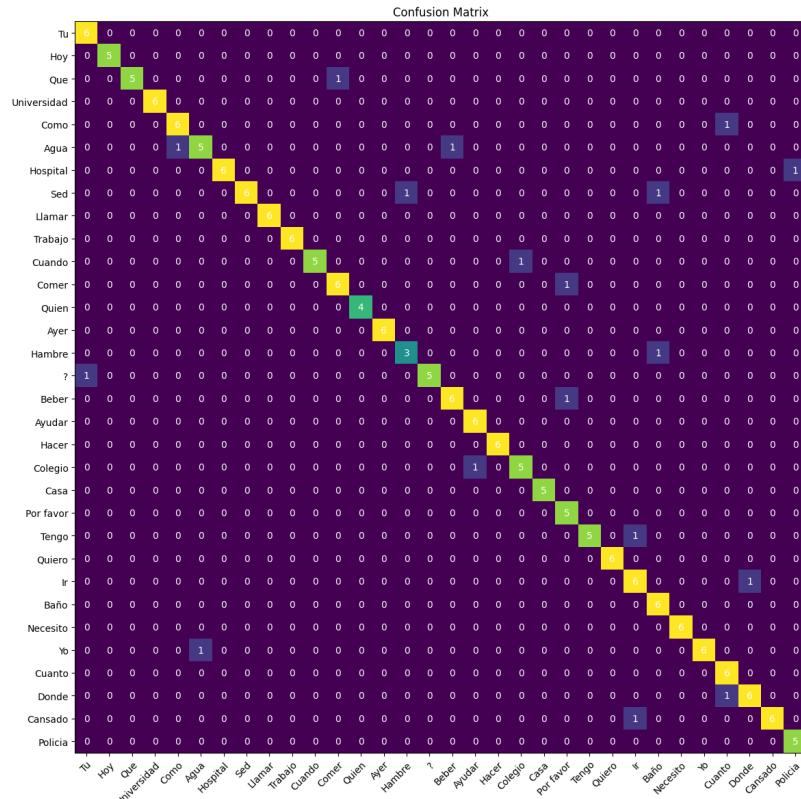


Figura 8.24: Matriz de confusión del modelo con normalización por lotes

8.2.6. Combinación de dropout y normalización por lotes

En esta iteración, se combinaron las técnicas de dropout y normalización por lotes, con el objetivo de evaluar si la combinación de ambas técnicas mejoraba el desempeño del modelo. Se realizan pruebas con dos modelos distintos, con el objetivo de encontrar la combinación de dropout y normalización por lotes que maximizara el desempeño del modelo. La composición del primero de los modelos se puede observar en la figura 8.25.

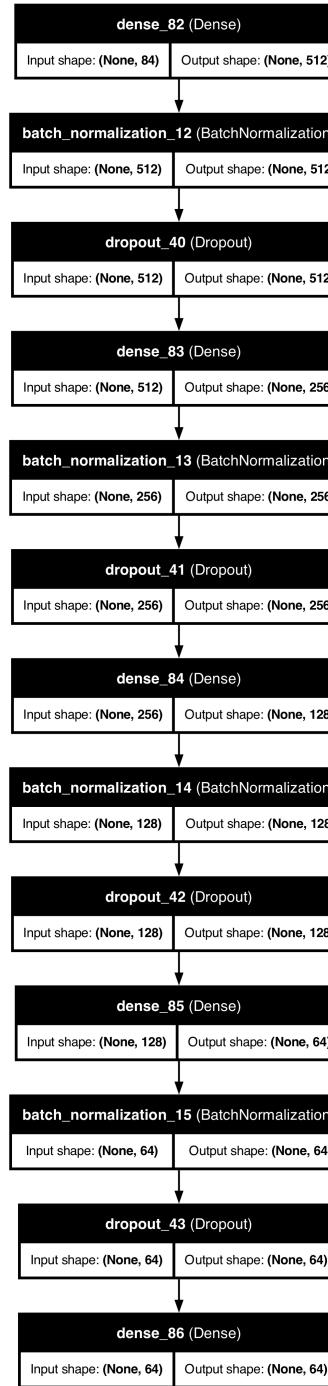


Figura 8.25: Primer modelo con dropout y normalización por lotes

El desempeño de este modelo se puede observar en la tabla 8.7. Este modelo tiene una precisión de 0.8593, una sensibilidad de 0.9461 y un F1 de 0.9441. Adicional a esto, se puede observar la matriz de confusión en la figura 8.27 y el historial de entrenamiento en la figura 8.26.

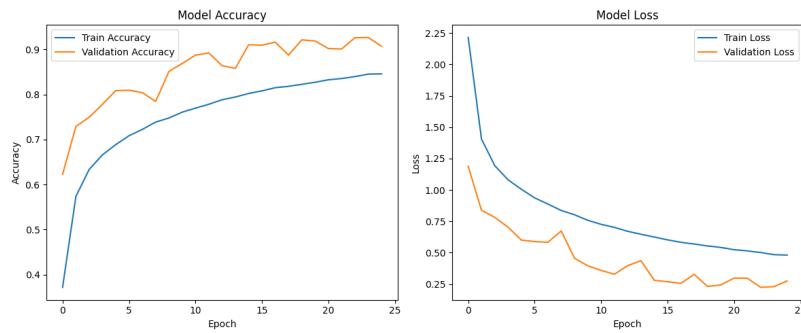


Figura 8.26: Historial de entrenamiento del primer modelo con dropout y normalización por lotes

Métrica	Valor
Precisión	0.8593
Sensibilidad	0.9461
F1	0.9441

Tabla 8.7: Desempeño del primer modelo con dropout y normalización por lotes

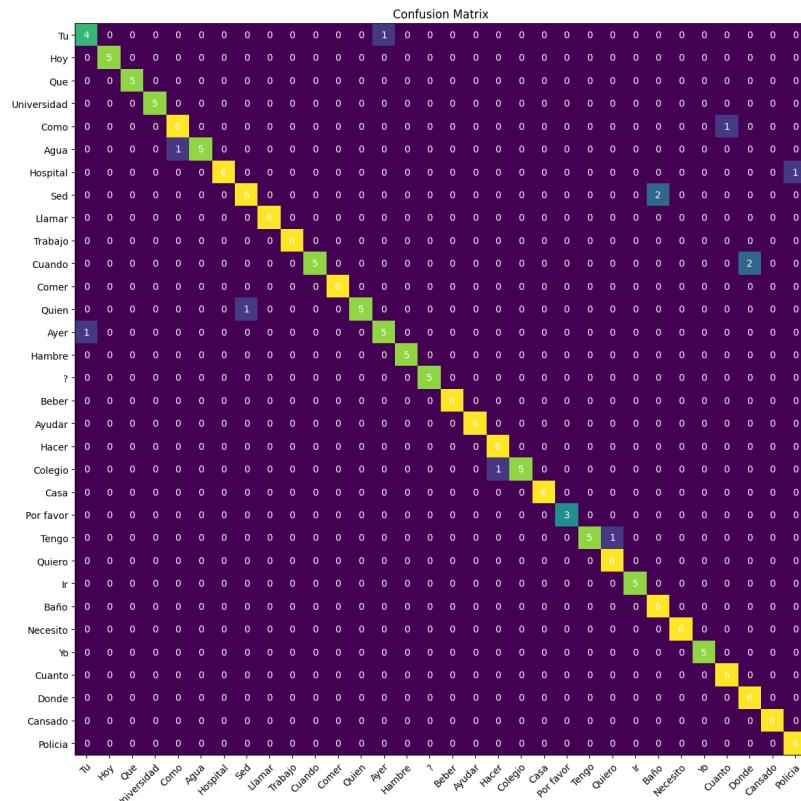


Figura 8.27: Matriz de confusión del primer modelo con dropout y normalización por lotes

La composición del segundo de los modelos se puede observar en la figura 8.28.

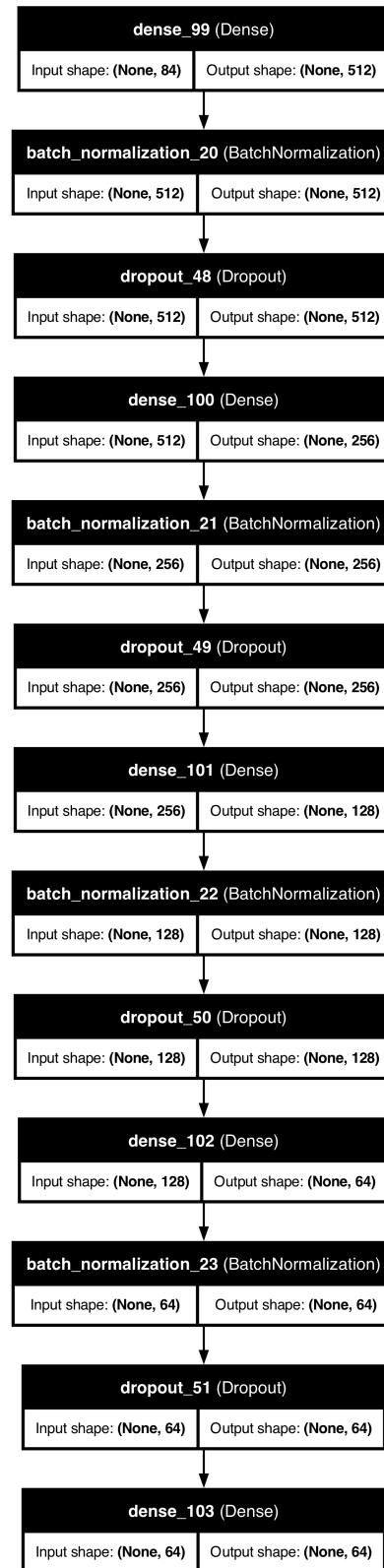


Figura 8.28: Segundo modelo con dropout y normalización por lotes

El desempeño de este modelo se puede observar en la tabla 8.8. Este modelo tiene una precisión de 0.8802, una sensibilidad de 0.9685 y un F1 de 0.9706. Adicional a esto, se puede observar la matriz de confusión en la figura 8.30 y el historial de entrenamiento en la figura 8.29.

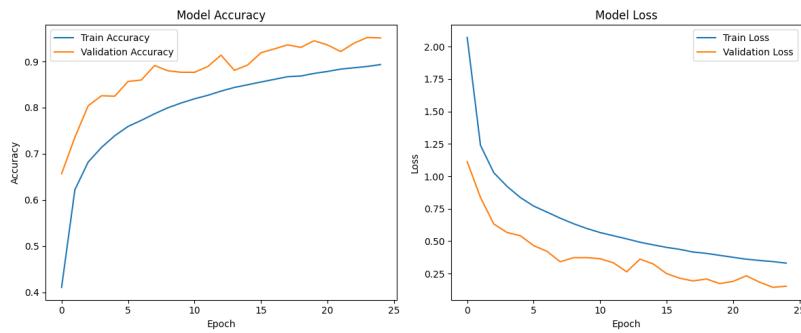


Figura 8.29: Historial de entrenamiento del segundo modelo con dropout y normalización por lotes

Métrica	Valor
Precisión	0.8802
Sensibilidad	0.9685
F1	0.9706

Tabla 8.8: Desempeño del segundo modelo con dropout y normalización por lotes

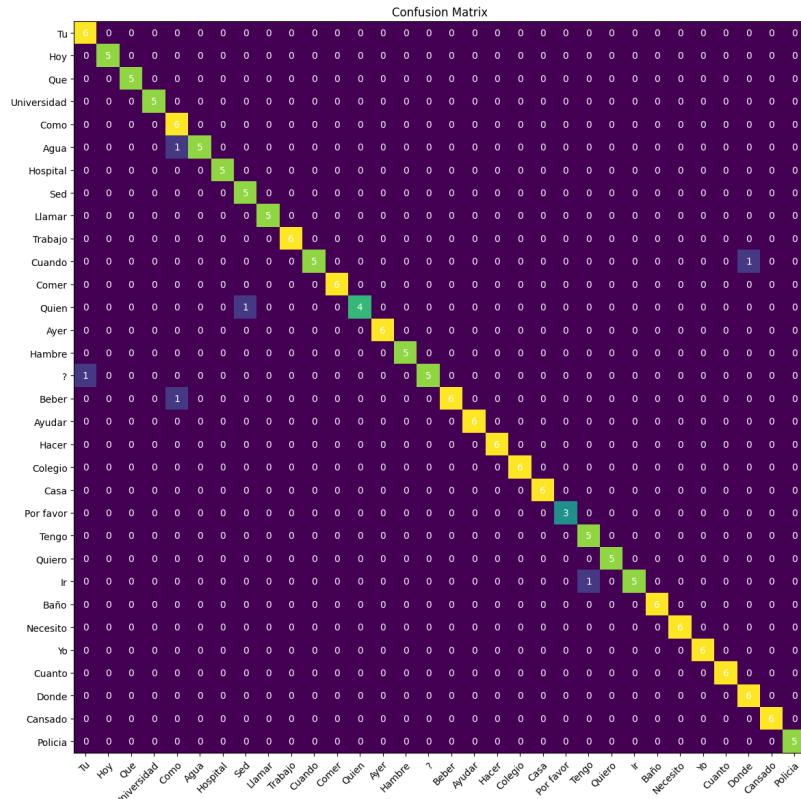


Figura 8.30: Matriz de confusión del segundo modelo con dropout y normalización por lotes

8.2.7. Elección del modelo final

El modelo final seleccionado es el modelo con dropout y normalización por lotes, con una precisión de 0.8802, una sensibilidad de 0.9685 y un F1 de 0.9706. Este modelo tiene un desempeño superior a los modelos anteriores, lo cual indica que la combinación de dropout y normalización por lotes es efectiva para mejorar el desempeño del modelo. Al observar la matriz de confusión en la figura 8.30, se puede observar que el modelo tiene un buen desempeño en la mayoría de las clases.

8.3. Evaluación del modelo

8.3.1. Reconocimiento de una palabra

El modelo final fue implementado en una aplicación de reconocimiento de lengua de señas de Guatemala en tiempo real, la cual se probó con un conjunto de datos de prueba. Este conjunto de datos de prueba está compuesto de 192 videos, los cuales contienen gestos de 32 palabras de la lengua de señas de Guatemala. Al utilizar el modelo final, se obtuvieron resultados satisfactorios, ya que el modelo fue capaz de reconocer correctamente los gestos de lengua de señas en tiempo real. En la figura 8.31 se puede observar el desempeño del modelo en tiempo real. Adicionalmente, se puede observar el desempeño del modelo en la tabla 8.9.

Métrica	Valor
Precisión	0.8802
Sensibilidad	0.9685
F1	0.9706
FPS promedio	19.38

Tabla 8.9: Desempeño del segundo modelo de la combinación de dropout y normalización por lotes

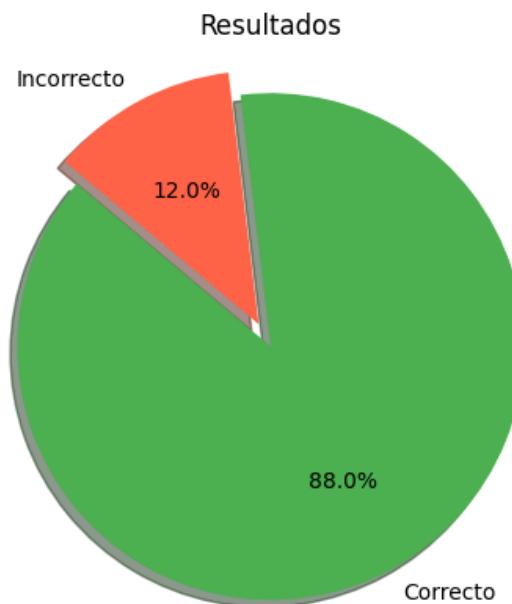


Figura 8.31: Reconocimiento de lengua de señas de Guatemala en tiempo real (una palabra)

8.3.2. Reconocimiento de múltiples palabras

Para evaluar el desempeño del modelo en la detección de múltiples palabras, se utilizó un conjunto de datos de prueba con videos de varias señas. Este conjunto de datos de prueba está compuesto de 27 videos, los cuales contienen gestos de 2 a 4 palabras de la lengua de señas de Guatemala. Al utilizar el modelo final, no se obtuvieron resultados satisfactorios, ya que el modelo no fue capaz de reconocer correctamente los gestos de lengua de señas en tiempo real. En la figura 8.32 se puede observar el desempeño del modelo en tiempo real.



Figura 8.32: Reconocimiento de lengua de señas de Guatemala en tiempo real (múltiples palabras)

8.4. Aplicaciones del modelo

8.4.1. Pruebas en tiempo real

En las pruebas realizadas en tiempo real, el modelo final fue capaz de reconocer correctamente los gestos de lengua de señas de Guatemala en la mayoría de los casos. Estas pruebas permitieron observar como el modelo se comporta en un entorno real, y si es capaz de reconocer todos los puntos clave de las señas. A continuación se muestran algunas capturas de pantalla de la aplicación de reconocimiento de lengua de señas de Guatemala en tiempo real. Cabe mencionar que cada uno de estos dos videos fueron grabados en diferentes angulos, con diferentes dispositivos, y con diferentes personas realizando los gestos. Las figuras 8.33, y 8.34 muestran el funcionamiento de la aplicación en tiempo real, con las palabras *ayer* y *comer*, respectivamente.



Figura 8.33: Reconocimiento de la palabra *ayer* en tiempo real

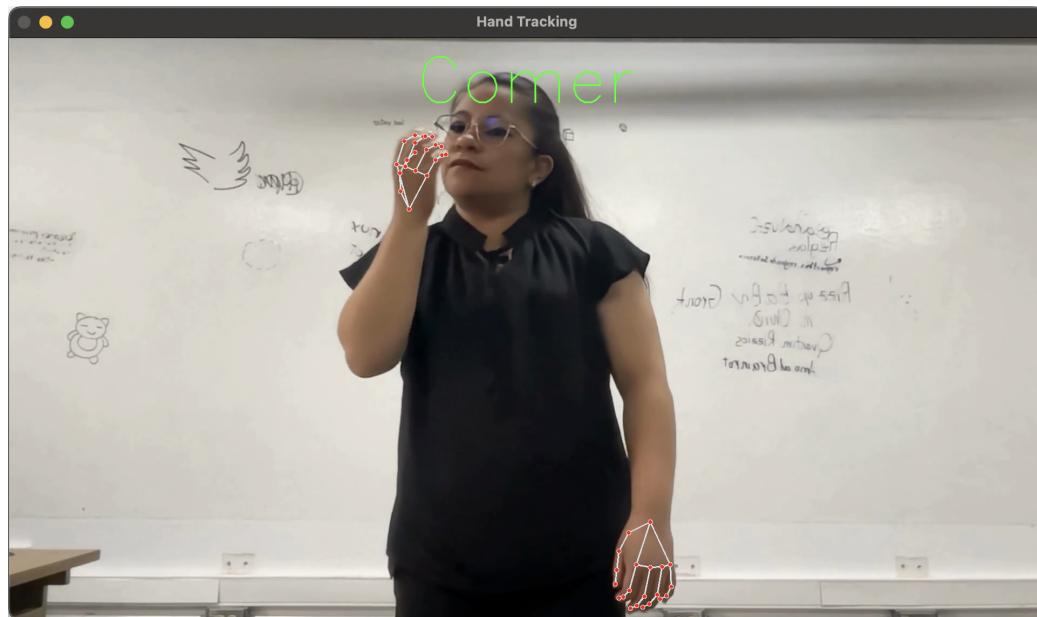


Figura 8.34: Reconocimiento de la palabra *comer* en tiempo real

8.4.2. Pruebas del API

El modelo final fue implementado en una Interfaz de Programación de Aplicaciones (API), la cual fue probada utilizando Postman y permitió acceder al modelo de reconocimiento de lengua de señas de forma remota. A continuación se muestran algunas capturas de pantalla de las pruebas realizadas con el API. Las figuras 8.35, y 8.36 muestran el funcionamiento del API, con las palabras *universidad* y *tu*, respectivamente.

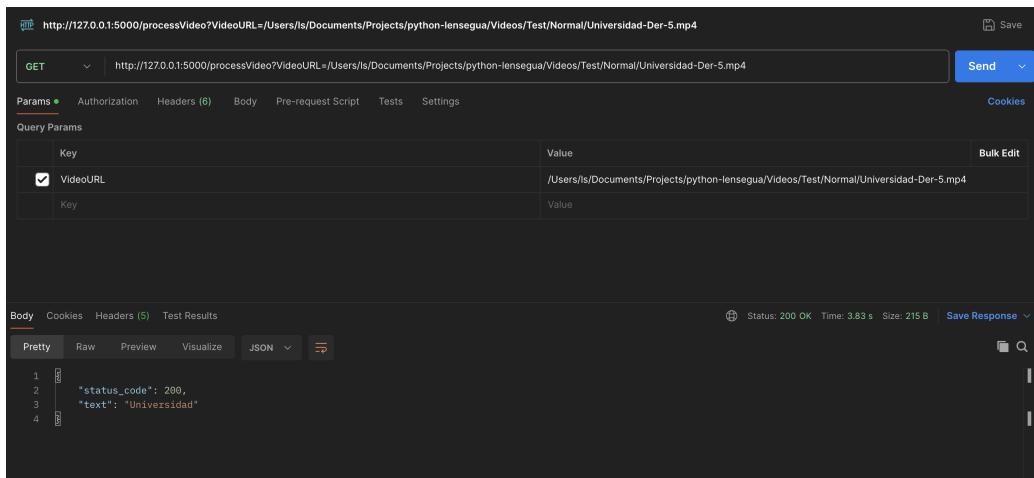


Figura 8.35: Reconocimiento de la palabra *universidad* utilizando el API

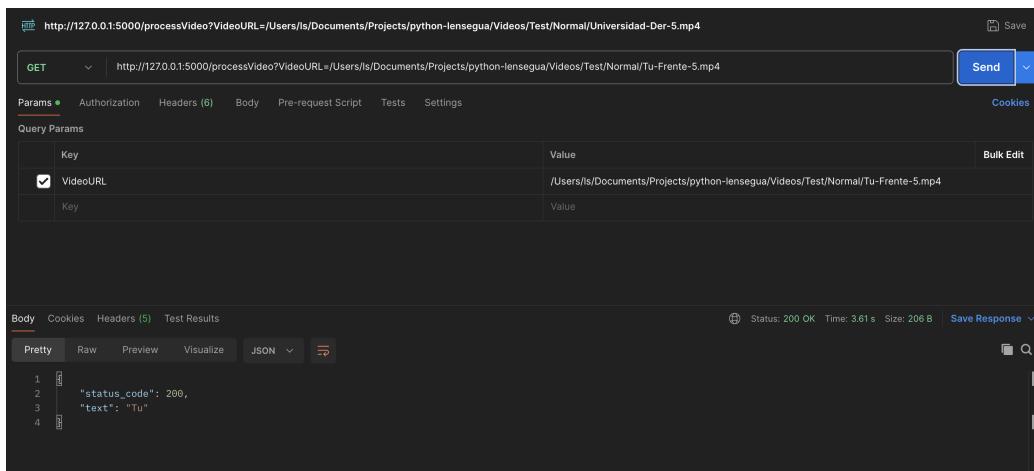


Figura 8.36: Reconocimiento de la palabra *tu* utilizando el API

CAPÍTULO 9

Análisis de resultados

9.1. Desempeño del modelo

El modelo final tiene una precisión de 0.8802, una sensibilidad de 0.9685 y una puntuación F1 de 0.9706. Esto indica que el modelo tiene un buen desempeño en la clasificación de los gestos de lengua de señas de Guatemala. Adicional a esto, el modelo tiene la capacidad de procesar 19.38 fotogramas por segundo, lo cual indica que el modelo es capaz de reconocer los gestos en tiempo real con una latencia baja. En otras palabras, para un video grabado a 30 fotogramas por segundo, la velocidad más comúnmente utilizada en dispositivos móviles, el modelo es capaz de procesar los fotogramas a un 64.6 % de la velocidad de grabación. Al utilizar un video grabado a 24 fotogramas por segundo, la velocidad relativa del modelo aumenta a un 80.75 % de la velocidad de grabación.

Para la elección del modelo final, se consideraron varios factores, como la precisión, la sensibilidad, y la puntuación F1. No se consideró la velocidad de procesamiento del modelo, ya que en general los modelos de reconocimiento de lengua de señas presentaron velocidades de procesamiento muy similares. La precisión, la cual indica la proporción de predicciones correctas, fue el factor más importante en la elección del modelo final, ya que se consideró que era importante que el modelo fuera capaz de reconocer correctamente los gestos de lengua de señas. Sin embargo, la sensibilidad y la puntuación F1 también fueron factores importantes, ya que indican la proporción de verdaderos positivos y la precisión del modelo, respectivamente.

Un modelo con una alta precisión, pero que produjera muchos falsos positivos o tuviera una baja sensibilidad, no sería útil en la práctica, ya que no sería capaz de reconocer correctamente los gestos de lengua de señas. En el caso de un modelo con baja sensibilidad, este modelo no sería capaz de predecir en la mayoría de los casos, lo cual resultaría en un modelo poco útil. Por otro lado, un modelo con una alta sensibilidad, pero que produjera muchos falsos positivos, tampoco sería útil en la práctica, ya que produciría mucho ruido en las predicciones. Balancear estos dos factores es importante para obtener un modelo con un buen desempeño en la clasificación de los gestos de lengua de señas, lo cual se logró con el modelo final.

En las pruebas realizadas en tiempo real, el modelo final fue capaz de reconocer correctamente los gestos de lengua de señas de Guatemala en la mayoría de los casos. De los 192 videos de prueba, el

modelo fue capaz de reconocer correctamente los gestos de lengua de señas en 169, lo cual corresponde a un 88 % de precisión. En la figura 8.31 se puede observar el desempeño del modelo en tiempo real. Sin embargo, en las pruebas de reconocimiento de múltiples palabras, el modelo no fue capaz de reconocer correctamente los gestos de lengua de señas. De los 27 videos de prueba, el modelo no fue capaz de reconocer correctamente los gestos de lengua de señas en ninguno de los casos.

9.2. Limitaciones del modelo

A pesar de que el modelo final tiene un buen desempeño en la clasificación de los gestos de lengua de señas de Guatemala, existen algunas limitaciones que deben ser consideradas. Una de las limitaciones del modelo es que no es capaz de reconocer correctamente los gestos de lengua de señas en tiempo real en todos los casos. En las pruebas realizadas con un conjunto de datos de una sola palabra por video, el modelo fue capaz de reconocer la seña correctamente en un 88 % de los casos. Sin embargo, en las pruebas de reconocimiento de múltiples palabras, el modelo no fue capaz de reconocer correctamente los gestos de lengua de señas en ninguno de los casos. Esto indica que el modelo tiene dificultades para reconocer los gestos de lengua de señas en tiempo real cuando se presentan múltiples palabras en un video.

Esta limitación se debe a varios factores, entre los cuales esta el conjunto de datos de entrenamiento y la forma en la que se identifica el inicio y el fin de una seña. Al realizar el análisis de similitud entre las clases, se observó que algunas clases tenían un alto grado de similitud en los movimientos de las manos, lo cual puede dificultar la distinción entre ellas. Estas clases que tienen un alto grado de similitud son las que presentan mayores dificultades para el modelo, lo cual lleva a que se produzcan errores en la clasificación de los gestos de lengua de señas. Adicional a esto, el análisis de similitud entre todas las clases mostró que la mayoría de las clases presentan cierto grado de superposición, lo cual puede dificultar la distinción entre ellas.

Esta limitación proveniente del conjunto de datos esta directamente relacionada con la forma en la que se identifica el inicio y el fin de una seña, la cual constituye otra limitación del modelo. Debido a que el modelo distingue el inicio y el fin de una seña utilizando un umbral de confianza y un tiempo mínimo de duración, es posible que el modelo no sea capaz de identificar correctamente el inicio y el fin de una seña en todos los casos. La superposición entre las clases dificulta la identificación del inicio de una seña, ya que durante la transición entre una seña y otra, el modelo no puede predecir con certeza cuál es la seña que se está realizando. Al no ser capaz de identificar correctamente la seña durante estos períodos de transición, el modelo puede producir errores en la clasificación de los gestos de lengua de señas. Esto lleva a un efecto de acumulación de errores, y hace que cada seña consecutiva sea más difícil de reconocer correctamente.

Como se mencionó anteriormente, el conjunto de datos de entrenamiento fue un factor limitante en el desempeño del modelo. Adicional a la superposición entre las clases, el conjunto de datos de entrenamiento también presentó desafíos en el balance de las clases. Debido a que el conjunto de datos está compuesto por los fotogramas de los videos, y no por los videos completos, es posible que algunas clases tengan más fotogramas que otras. Esto sucede con las señas que tienen duración más larga, ya que estas señas tienen más fotogramas en el conjunto de datos. Al tener más fotogramas, estas clases tienen una mayor probabilidad de ser seleccionadas durante el entrenamiento, lo cual puede llevar a un desbalance en el conjunto de datos.

El desbalance en el conjunto de datos puede llevar a que el modelo tenga dificultades para reconocer las clases minoritarias, ya que estas clases tienen menos ejemplos en el conjunto de datos. Esto se podría solucionar utilizando técnicas de disminución de la clase mayoritaria, como la de submuestreo, pero esto podría llevar a una pérdida de información en el conjunto de datos. Debido a que no se puede seleccionar los fotogramas a eliminar del conjunto de datos para el submuestreo, es posible que se eliminen fotogramas importantes para la clasificación de los gestos de lengua de

señas. La eliminación de estos fotogramas podría llevar a una pérdida de información en el conjunto de datos, lo cual podría llevar a un desempeño inferior del modelo.

La forma correcta de lidiar con el desbalance en el conjunto de datos es desde la recolección de los datos, ya que no se puede solucionar completamente en la etapa de entrenamiento. Al recolectar los datos, es importante asegurarse de que todas las clases tengan una cantidad similar total de fotogramas en el conjunto de datos. Para este proyecto, el enfoque de la recolección de datos fue recolectar la misma cantidad de videos para cada clase, pero no se consideró la duración de los videos. Al considerar la duración de los videos, se podrían haber recolectado más videos para las clases que tienen una duración más corta, y menos videos para las clases que tienen una duración más larga. Esto habría permitido tener un conjunto de datos más balanceado, y habría mejorado el desempeño del modelo en la clasificación de las señas.

9.3. Cumplimiento de los objetivos

El objetivo principal de este proyecto era desarrollar un modelo de reconocimiento de lengua de señas de Guatemala en tiempo real, el cual fuera capaz de reconocer un conjunto establecido de palabras. Este objetivo se logró, ya que el modelo final fue capaz de reconocer correctamente las palabras de lengua de señas de Guatemala en las pruebas realizadas en tiempo real, con una precisión de 0.8802, una sensibilidad de 0.9685 y una puntuación F1 de 0.9706. Adicional a esto, se creó un proyecto de código abierto en GitHub, el cual promueve el desarrollo de futuras investigaciones en el campo del reconocimiento de lengua de señas.

Adicional al objetivo principal, se plantearon varios objetivos específicos, la mayoría de los cuales se lograron. El primer objetivo específico era recolectar un conjunto de datos de lengua de señas de Guatemala, el cual se logró con la recolección de 960 videos de 32 palabras de la lengua de señas de Guatemala. El segundo objetivo específico era preprocessar el conjunto de datos, el cual se logró con la edición, el etiquetado manual de los videos y la extracción de los fotogramas de los videos. El tercer objetivo específico era crear un sistema con la capacidad de reconocer los puntos clave de las señas, el cual se logró con la implementación de aplicaciones de reconocimiento de lengua de señas en tiempo real. El cuarto objetivo específico era entrenar un modelo de reconocimiento de lengua de señas de Guatemala, el cual se logró con la implementación de un modelo de redes neuronales, el cual fue entrenado con el conjunto de datos recolectado.

El quinto objetivo específico era diseñar un sistema de reconocimiento de lengua de señas de Guatemala en tiempo real, el cual fuera capaz de identificar el inicio y el fin de una señal, con el objetivo de reconocer múltiples palabras en un video. Este objetivo no se logró completamente, ya que el modelo no fue capaz de reconocer correctamente los gestos de lengua de señas en tiempo real cuando se presentaban múltiples palabras en un video. Debido a las limitaciones del modelo, el sistema no fue capaz de identificar correctamente el inicio y el fin de las señas, lo cual llevó a errores acumulativos en la clasificación de las señas. A pesar de esto, el proyecto logró cumplir con la mayoría de los objetivos planteados, y logró desarrollar un modelo de reconocimiento de lengua de señas de Guatemala en tiempo real.

CAPÍTULO 10

Conclusiones

El desarrollo de un modelo de reconocimiento de lengua de señas de Guatemala ha demostrado ser un esfuerzo significativo en la comprensión y la implementación de tecnologías de inteligencia artificial en el ámbito de la comunicación. A través de un proceso iterativo de recolección de datos, modelado y evaluación, se logró un modelo final que presenta una precisión de 0.8802, una sensibilidad de 0.9685 y un F1 de 0.9706, lo cual indica un desempeño aceptable en la clasificación de gestos de lengua de señas en tiempo real.

A pesar de estos logros, el modelo enfrenta limitaciones significativas, especialmente en su capacidad para reconocer correctamente múltiples palabras en un solo video. Estas limitaciones se derivan en gran medida del desbalance en el conjunto de datos y de la dificultad en la identificación del inicio y fin de las señas. Reconocer y abordar estas limitaciones es crucial para el futuro desarrollo y mejora de este tipo de tecnología.

La implementación exitosa del modelo en una aplicación de reconocimiento de lengua de señas de Guatemala y en un API muestra el potencial de esta tecnología para facilitar la comunicación y promover la inclusión de las personas sordas en diferentes contextos. Con las recomendaciones planteadas, se espera que futuras iteraciones de este proyecto no solo mejoren la precisión y la eficiencia del modelo, sino que también contribuyan a una mayor accesibilidad en el uso de la lengua de señas de Guatemala en la vida cotidiana.

Para finalizar, este proyecto no solo resalta el valor de la inteligencia artificial en el reconocimiento de la lengua de señas, sino que también abre la puerta a futuras investigaciones y desarrollos en este campo, con el potencial de transformar la manera en que se comunican las personas sordas y oyentes en Guatemala.

CAPÍTULO 11

Recomendaciones

Basado en la elaboración de este proyecto, y con el fin de permitir la realización de futuros proyectos similares a este, se presentan las siguientes recomendaciones:

1. **Recolección de Datos Balanceada:** Para mejorar el desempeño del modelo, se recomienda realizar una recolección de datos más equilibrada, teniendo en cuenta la duración de las señas. Además, asegurarse de que cada clase tenga un número similar de fotogramas en el conjunto de datos puede ayudar a mitigar el problema del desbalance y mejorar la capacidad del modelo para reconocer gestos de clases minoritarias.
2. **Ampliación del Conjunto de Datos:** Se sugiere ampliar el conjunto de datos con más ejemplos de cada clase, especialmente para las que presentan una mayor confusión en el modelo. Esto puede incluir la recopilación de más videos de señas en diferentes contextos y variaciones para aumentar la diversidad y robustez del conjunto de datos.
3. **Mejorar la Identificación de Inicio y Fin de Señas:** Se recomienda explorar técnicas alternativas para identificar el inicio y fin de las señas en los videos, como el uso de una segunda red neuronal para detectar estos puntos clave.
4. **Técnicas de Aumento:** Implementar técnicas de aumento de datos, como la rotación, la variación en la iluminación y la manipulación de la velocidad de los videos, puede ayudar a mejorar la generalización del modelo al introducir variaciones en los gestos que el modelo debe aprender a reconocer.
5. **Entrenamiento en Diferentes Condiciones:** Realizar pruebas y entrenamientos en diferentes entornos y condiciones, como diferentes ángulos, iluminación y antecedentes, puede proporcionar una visión más robusta del desempeño del modelo y permitir su adaptación a situaciones de la vida real.
6. **Colaboración con Expertos en Lengua de Señas:** Trabajar en colaboración con intérpretes y expertos en lengua de señas puede ayudar a identificar signos que son comúnmente confundidos y proporcionar una mejor comprensión de las variaciones en el uso de la lengua de señas de Guatemala.

Bibliografía

- [1] Aakanksha, N.: *Image Processing and Data Augmentation Techniques for Computer Vision*, 2020. Recuperado el 30 de mayo de 2024, de <https://towardsdatascience.com/image-processing-techniques-for-computer-vision-11f92f511e21>.
- [2] Altunay, N.: *The basics of image processing and OpenCV*, 2019. Recuperado el 30 de mayo de 2024, de <https://developer.ibm.com/articles/learn-the-basics-of-computer-vision-and-object-detection/#before-everything-what-is-opencv-10>.
- [3] Blanchfield, B., J. Feldman, J. Dunbar y E. Gardner: *The severely to profoundly hearing-impaired population in the United States: Prevalence estimates and demographics*. Journal of the American Academy of Audiology, páginas 183–189, 2001. Recuperado el 23 de mayo de 2024.
- [4] Bos, S.: *What Are Decibels, and How Are They Measured?*, 2023. Recuperado el 23 de mayo de 2024, de <https://science.howstuffworks.com/question124.htm>.
- [5] CONADI: *Acuerdo gubernativo número 121-2021*, 2021. Recuperado el 5 de febrero de 2024, de <https://www.conadi.gob.gt/ley-de-los-derechos-de-las-personas-con-discapacidad/>.
- [6] Forsyth, D. y J. Ponce: *Computer vision: A modern approach*, 2012. Recuperado el 30 de mayo de 2024, de <https://archive.org/details/computervisionmo0000fors/page/696/mode/2up>.
- [7] Foundation, Hearing Health: *What Are Safe Decibels?*, n.d. Recuperado el 23 de mayo de 2024, de <https://hearinghealthfoundation.org/keeplistenning/decibels>.
- [8] Goldsmisth, J.: *The Value of Computer Vision in Healthcare*, 2021. Recuperado el 30 de mayo de 2024, de <https://www.himss.org/resources/value-computer-vision-healthcare>.
- [9] Goodfellow, I., Y. Bengio y A. Courville: *Deep learning*. MIT Press, 2016. Recuperado el 31 de mayo de 2024, de <https://link.springer.com/article/10.1007/s10710-017-9314-z>.
- [10] Google: *On-device machine learning for everyone*, n.d. Recuperado el 21 de abril de 2024, de <https://developers.google.com/mediapipe>.
- [11] Google: *On-device machine learning for everyone*, n.d. Recuperado el 21 de abril de 2024, de <https://developers.google.com/mediapipe>.
- [12] Guatemala, Congreso de la república de: *Ley que Reconoce y Aprueba la Lengua de Señas de Guatemala, LENSEGUÁ*, 2020. Recuperado el 23 de mayo de 2024,

- de <http://ww2.oj.gob.gt/es/queesoj/estructuraoj/unidadesadministrativas/centroanalisisdocumentacionjudicial/cds/cds%20de%20leyes/2020/pdfs/decretos/D03-2020.pdf>.
- [13] IBM: *What is a neural network?*, n.d. Recuperado el 31 de mayo de 2024, de <https://www.ibm.com/topics/neural-networks>.
 - [14] IBM: *¿Qué es una matriz de confusión?*, n.d. Recuperado el 20 de octubre de 2024, de <https://www.ibm.com/mx-es/topics/confusion-matrix>.
 - [15] Journey, ML: *What is Recall in Machine Learning?*, 2024. Recuperado el 20 de octubre de 2024, de <https://mljourney.com/what-is-recall-in-machine-learning/>.
 - [16] Kucsván, Z.: *Comparing two- and three-view computer vision*, 2019. Recuperado el 23 de abril de 2024, de https://www.researchgate.net/publication/335322802_Comparing_two-_and_three-view_computer_vision.
 - [17] Kundu, R.: *F1 Score in Machine Learning: Intro and Calculation*, 2022. Recuperado el 20 de octubre de 2024, de <https://www.v7labs.com/blog/f1-score-guide>.
 - [18] López, J.: *Teorema de Aproximación Universal*. Tesis de Doctorado, Universidad del Valle de Guatemala, 2023.
 - [19] Medicina, Biblioteca Nacional de: *Impact of Hearing Loss on Daily Life and the Workplace*, 2005. Recuperado el 23 de mayo de 2024, de <https://www.ncbi.nlm.nih.gov/books/NBK207836/>.
 - [20] OpenCV: *OpenCV: Introduction*, n.d.
 - [21] Salud, Organización Mundial de la: *Deafness and hearing loss*, 2018. Recuperado el 18 de mayo de 2024, de <https://www.who.int/news-room/fact-sheets/detail/deafness-and-hearing-loss>.

