



**UNIVERSIDAD DE GUAYAQUIL
FACULTAD DE INGENIERÍA INDUSTRIAL
CARRERA DE INGENIERÍA EN TELEINFORMÁTICA**

**TRABAJO DE TITULACIÓN
PREVIO A LA OBTENCIÓN DEL TÍTULO DE
INGENIERO EN TELEINFORMÁTICA**

**ÁREA
TECNOLOGÍAS APLICADAS**

**TEMA
“TRADUCTOR DE LENGUAJE DE SEÑAS BASADO EN
VISIÓN ARTIFICIAL PARA PERSONAS CON
DISCAPACIDAD AUDITIVA.”**

**AUTOR
ALARCÓN MONSERRATE NELSON ALEXANDER**

**DIRECTOR DEL TRABAJO
ING. COMP. CASTILLO LEÓN ROSA ELIZABETH, MG.**

GUAYAQUIL, SEPTIEMBRE 2022



**ANEXO XI.- FICHA DE REGISTRO
DE TRABAJO DE TITULACIÓN
FACULTAD INGENIERÍA INDUSTRIAL
CARRERA DE INGENIERÍA EN TELEINFORMÁTICA**



REPOSITORIO NACIONAL EN CIENCIA Y TECNOLOGÍA			
FICHA DE REGISTRO DE TRABAJO DE TITULACIÓN			
TÍTULO:	Traductor de Lenguaje de señas basado en Visión Artificial para personas con discapacidad auditiva		
AUTOR (apellidos y nombres):	Alarcón Monserrate Nelson Alexander.		
TUTOR y REVISOR (apellidos y nombres):	Ing. Comp. Castillo León Rosa Elizabeth, Mg. Ing. Sist. García Torres Ingrid Angélica, Mg.		
INSTITUCIÓN:	Universidad de Guayaquil		
UNIDAD/FACULTAD:	Facultad de Ingeniería Industrial		
MAESTRÍA/ESPECIALIDAD :			
GRADO OBTENIDO:	Ingeniero en Teleinformática		
FECHA DE PUBLICACIÓN:	28 de septiembre del 2022	No. DE PÁGINAS:	129
ÁREAS TEMÁTICAS:	Tecnologías Aplicadas		
PALABRAS CLAVES/ KEYWORDS:	Visión Artificial, Discapacidad auditiva, Jetson Nano, Lengua de señas, Python, Red Neuronal Convolucional.		
<p>Resumen</p> <p>El objetivo principal de esta investigación es realizar un prototipo de interpretación y traducción de gestos (únicamente letras) mediante visión artificial y Python. El desarrollo del prototipo empieza con la recolección de Data en la que se utilizaron aproximadamente 13000 imágenes para la creación de un Dataset propio. Lo siguiente fue el desarrollo de una red neuronal convolucional (RNC) que dará como resultado un modelo de entrenamiento basado en el Dataset previo. Posteriormente se traslada el sistema traductor de seña al prototipo Jetson Nano donde se ejecutará en un entorno virtual que contenga las librerías específicas para su funcionamiento. Finalmente, se muestran las pruebas realizadas en distintos escenarios donde se comprobó el funcionamiento del prototipo, notando que las inferencias alcanzaban un 80% de efectividad en pantalla. La importancia del prototipo radica en que mediante nuevas tecnologías y visión artificial se puede ayudar a toda una comunidad y fomentar la inclusión.</p>			

Abstract

The main objective of this research is to make a prototype of interpretation and translation of gestures (only letters) using artificial vision and Python. The development of the prototype begins with the collection of data in which approximately 13,000 images were used to create its own dataset. The next step was the development of a convolutional neural network (CNN) that will result in a training model based on the previously created Dataset. Subsequently, the sign translator system is transferred to the Jetson Nano prototype where it will be executed in a virtual environment that contains the specific libraries for its correct operation. Finally, the tests carried out in different scenarios where the functioning of the prototype was verified, noting that the inferences reached 80% effectiveness on screen. The importance of the prototype lies in the fact that through new technologies and artificial vision an entire community can be helped and at the same time inclusion can be promoted.

ADJUNTO PDF:	SI (X)	NO
CONTACTO CON AUTOR/ES:	Teléfono: 0998030264	E-mail: nelson.alarconm@ug.edu.ec
CONTACTO CON LA INSTITUCIÓN:	Nombre: Ing. Ramón Maquilón Nicola, MG.	
	Teléfono: 593-2658128	
	E-mail: direccionti@ug.edu.ec	



**ANEXO XII.- DECLARACIÓN DE AUTORÍA Y DE
AUTORIZACIÓN DE LICENCIA GRATUITA
INTRANSFERIBLE Y NO EXCLUSIVA PARA EL USO
NO COMERCIAL DE LA OBRA CON FINES
NO ACADÉMICOS
FACULTAD INGENIERÍA INDUSTRIAL**



CARRERA INGENIERÍA EN TELEINFORMÁTICA

**LICENCIA GRATUITA INTRANSFERIBLE Y NO COMERCIAL DE LA OBRA CON
FINES NO ACADÉMICOS**

Yo **ALARCÓN MONSERRATE NELSON ALEXANDER** con C.C. No. **0955187760** certifico que los contenidos desarrollados en este trabajo de titulación, cuyo título es **“TRADUCTOR DE LENGUAJE DE SEÑAS BASADO EN VISIÓN ARTIFICIAL PARA PERSONAS CON DISCAPACIDAD AUDITIVA.”** son de mi absoluta propiedad y responsabilidad, en conformidad al Artículo 114 del CÓDIGO ORGÁNICO DE LA ECONOMÍA SOCIAL DE LOS CONOCIMIENTOS, CREATIVIDAD E INNOVACIÓN*, autorizo la utilización de una licencia gratuita intransferible, para el uso no comercial de la presente obra a favor de la Universidad de Guayaquil.

A handwritten signature in black ink, appearing to read "Alarcón", written over a horizontal line.

ALARCÓN MONSERRATE NELSON ALEXANDER
C.C.: 0955187760



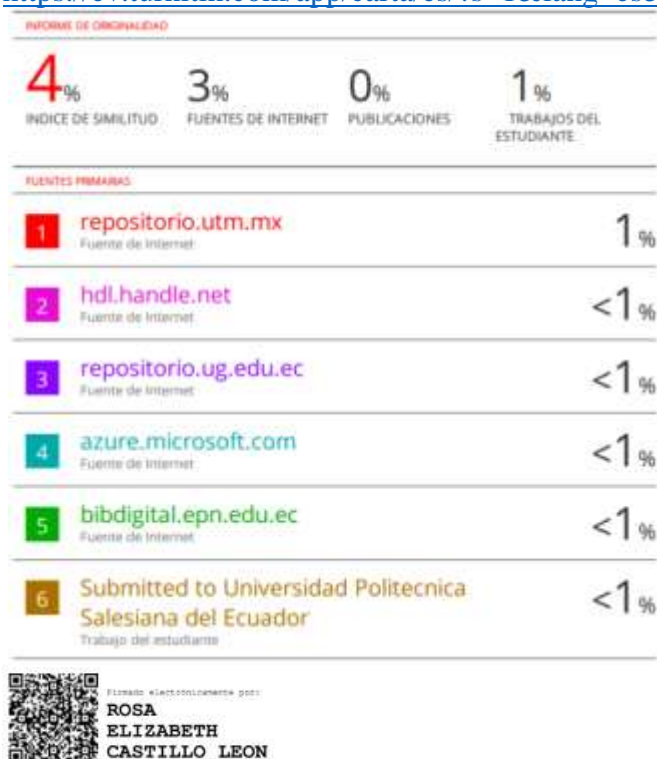
**ANEXO VII.- CERTIFICADO PORCENTAJE DE SIMILITUD
FACULTAD INGENIERÍA INDUSTRIAL CARRERA
INGENIERÍA EN TELEINFORMÁTICA**



Habiendo sido nombrado **ING. COMP. CASTILLO LEÓN ROSA ELIZABETH, MG**, tutor del trabajo de titulación certifico que el presente trabajo de titulación ha sido elaborado por **ALARCÓN MONSERRATE NELSON ALEXANDER**, con mi respectiva supervisión como requerimiento parcial para la obtención del título de **INGENIERO EN TELEINFORMÁTICA**.

Se informa que el trabajo de titulación: **TRADUCTOR DE LENGUAJE DE SEÑAS BASADO EN VISIÓN ARTIFICIAL PARA PERSONAS CON DISCAPACIDAD AUDITIVA**, ha sido orientado durante todo el periodo de ejecución en el programa Antiplagio TURNITIN quedando el 4% de coincidencia.

<https://ev.turnitin.com/app/carta/es/?s=1&lang=es&o=1895571645&u=1133714545>



ING. COMP. CASTILLO LEÓN ROSA ELIZABETH, MG.
DOCENTE TUTOR
C.C.: 0922372610

FECHA: 12/09/2022



**ANEXO VI. - CERTIFICADO DEL DOCENTE-TUTOR
DEL TRABAJO DE TITULACIÓN
FACULTAD INGENIERÍA INDUSTRIAL
CARRERA INGENIERÍA EN TELEINFORMÁTICA**



Guayaquil, 12 de septiembre de 2022,

Sr (a).

Ing. Annabelle Lizarzaburu Mora, MG.

Director(a) de Carrera Ingeniería en Teleinformática / Telemática

**FACULTAD DE INGENIERÍA INDUSTRIAL DE LA UNIVERSIDAD DE
GUAYAQUIL**

Ciudad. –

De mis consideraciones:

Envío a Ud. el Informe correspondiente a la tutoría realizada al Trabajo de Titulación **TRADUCTOR DE LENGUAJE DE SEÑAS BASADO EN VISIÓN ARTIFICIAL PARA PERSONAS CON DISCAPACIDAD AUDITIVA** del estudiante **ALARCÓN MONSERRATE NELSON ALEXANDER**, indicando que ha cumplido con todos los parámetros establecidos en la normativa vigente:

- El trabajo es el resultado de una investigación.
- El estudiante demuestra conocimiento profesional integral.
- El trabajo presenta una propuesta en el área de conocimiento.
- El nivel de argumentación es coherente con el campo de conocimiento.

Adicionalmente, se adjunta el certificado de porcentaje de similitud y la valoración del trabajo de titulación con la respectiva calificación.

Dando por concluida esta tutoría de trabajo de titulación, **CERTIFICO**, para los fines pertinentes, que el estudiante está apto para continuar con el proceso de revisión final.

Atentamente,



Firmado electrónicamente por:

**ROSA
ELIZABETH
CASTILLO LEON**

**ING. COMP. CASTILLO LEÓN ROSA ELIZABETH, MG.
TUTOR DE TRABAJO DE TITULACIÓN
C.C.: 0922372619**

FECHA: 12 DE SEPTIEMBRE DE 2022



**ANEXO VIII.- INFORME DEL DOCENTE REVISOR
FACULTAD INGENIERÍA INDUSTRIAL
CARRERA INGENIERÍA EN TELEINFORMÁTICA**



Guayaquil, 22 de septiembre de 2022

Sr(a).

Ing. Annabelle Sally Lizaraburu Mora

Director (a) de Carrera Ingeniería en Teleinformática / Telemática

FACULTAD DE INGENIERÍA INDUSTRIAL DE LA UNIVERSIDAD DE GUAYAQUIL

Ciudad. –

De mis consideraciones:

Envío a Ud. el informe correspondiente a la REVISIÓN FINAL del Trabajo de Titulación **“TRADUCTOR DE LENGUAJE DE SEÑAS BASADO EN VISIÓN ARTIFICIAL PARA PERSONAS CON DISCAPACIDAD AUDITIVA.”**, del estudiante **ALARCÓN MONSERRATE NELSON ALEXANDER**. Las gestiones realizadas me permiten indicar que el trabajo fue revisado considerando todos los parámetros establecidos en las normativas vigentes, en el cumplimiento de los siguientes aspectos:

Cumplimiento de requisitos de forma:

El título tiene un máximo de 14 palabras.

La memoria escrita se ajusta a la estructura establecida.

El documento se ajusta a las normas de escritura científica seleccionadas por la Facultad.

La investigación es pertinente con la línea y sublíneas de investigación de la carrera.

Los soportes teóricos son de máximo 5 años.

La propuesta presentada es pertinente.

Cumplimiento con el Reglamento de Régimen Académico:

El trabajo es el resultado de una investigación.

El estudiante demuestra conocimiento profesional integral.

El trabajo presenta una propuesta en el área de conocimiento.

El nivel de argumentación es coherente con el campo de conocimiento.

Adicionalmente, se indica que fue revisado, el certificado de porcentaje de similitud, la valoración del tutor, así como de las páginas preliminares solicitadas, lo cual indica el que el trabajo de investigación cumple con los requisitos exigidos.

Una vez concluida esta revisión, considero que el estudiante está apto para continuar el proceso de titulación. Particular que comunicamos a usted para los fines pertinentes.

Atentamente,



Firmado electrónicamente por:
**INGRID
ANGELICA
GARCIA TORRES**

ING. SIST. GARCÍA TORRES INGRID ANGÉLICA, MG.

C.C.: 1308497682

FECHA: 23 de septiembre del 2022

Dedicatoria

El desarrollo de esta investigación está dedicado a las personas más influyentes para mí. En primer lugar, en memoria de las mujeres más importantes que tuve en mi vida, que son mi abuela Fervia y madre Vicky pues fueron fuertes, valientes y las amaré siempre. Al amor de mi vida Suiying pues no solo me ha visto crecer, sino que siempre me ha apoyado en todo lo que me he propuesto y ha estado para mí siempre. A la familia Morán -Chang (Sr. Ricardo, Sra. Olga, Francisco y Leew) por darme la bienvenida a su familia y apoyo incondicional en absolutamente todo a lo largo de la carrera. Finalmente, mis padres Iván y Daisy por siempre creer y estar orgullosos de mí.

Agradecimiento

A mi tutora Ing. Rosa Elizabeth Castillo León que, gracias a su tiempo, consejos, correcciones y enseñanzas, hoy puedo culminar con éxito este trabajo de titulación.

A mis compañeros más cercanos de la universidad (Jean, Israel, Erwin, Jaime y Dayanna) pues me brindaron su ayuda y apoyo a lo largo de la carrera.

Índice General

N°	Descripción	Pág.
	Introducción	1

Capítulo I El problema

N°	Descripción	Pág.
1.1	Planteamiento del problema	2
1.2	Formulación del problema de investigación	3
1.3	Justificación e importancia	3
1.4	Objetivos de la investigación	4
1.4.1	Objetivo general	4
1.4.2	Objetivos específicos	4
1.5	Alcance del proyecto	5
1.5.1	Delimitación física	5
1.5.2	Alcance	5

Capítulo II Marco Teórico

N°	Descripción	Pág.
2.1	Antecedentes del problema	6
2.2	Fundamentación teórica	9
2.2.1	Personas con discapacidad auditiva	9
2.2.2	Lengua de señas	10
2.2.3	Inteligencia artificial	11
2.2.4	Visión artificial	12
2.2.4.1	Campos de aplicación de visión artificial	13
2.2.5	Machine learning	13
2.2.5.1	Técnicas de machine learning	14
2.2.6	Redes neuronales artificiales	16
2.2.6.1	Tipos de redes neuronales artificiales	18
2.2.7	Redes neuronales convolucionales	21
2.2.7.1	Etapas convolucionales de las redes neuronales convolucionales	22
2.2.7.2	Etapas de clasificación de las redes neuronales convolucionales	27

N°	Descripción	Pág.
2.2.8	Software utilizado	28
2.2.8.1	Software	28
2.2.8.2	Software libre	28
2.2.8.3	Lenguajes de programación	29
2.2.8.4	Librerías para visión artificial y aprendizaje automatizado	33
2.2.8.5	Jetpack jetson nano	35
2.2.8.6	Raspberry pi os	36
2.2.9	Hardware	36
2.2.9.1	Jetson nano developer kit	36
2.2.9.2	Raspberry pi	37
2.2.9.3	Cámaras web	38
2.2.10	Dactilología de lengua de señas americano	39
2.3	Marco conceptual	40
2.3.1	Dactilología	40
2.3.2	Falttening	40
2.3.3	redes neuronales convolucionales	41
2.3.4	pooling	41
2.3.5	Python	42
2.3.6	Visión artificial	42
2.4	Marco legal	42
2.4.1	Constitución de la república del ecuador	42
2.4.2	Ley orgánica de discapacidades	43
2.4.3	TIC's y su uso para personas con discapacidad	44

Capítulo III

Metodología

N°	Descripción	Pág.
3.1	Metodología del proyecto	45
3.1.1	Modalidad de la investigación	45
3.1.2	Instrumentos de investigación	46
3.1.2.1	Bibliográfico	46
3.1.2.2	Experimentación	46
3.1.2.3	Encuesta	46

N°	Descripción	Pág.
3.2	Análisis de factibilidad	46
3.2.1	Factibilidad legal	46
3.2.2	Factibilidad técnica	47
3.2.3	Factibilidad económica	49
3.2.4	Factibilidad operacional	50
3.3	Población y muestra	51
3.3.1	Población	51
3.3.2	Muestra	51
3.4	Interpretación y análisis de datos	52
3.5	Desarrollo de la propuesta	60
3.5.1	Requerimientos funcionales	60
3.5.2	Requerimientos no funcionales	62
3.5.3	Entidades del sistema	62
3.5.4	Diagramas de casos de usos	63
3.5.4.1	Diagrama de caso de uso reconocimiento de manos	63
3.5.4.2	Diagrama de caso de uso seguimiento de manos	64
3.5.4.3	Diagrama de caso de uso predicción de seña	64
3.6	Desarrollo del prototipo	64
3.6.1	Creación del dataset del abecedario dactilológico	64
3.6.2	Estandarización de las imágenes	65
3.6.3	Preparación de los conjuntos de testeo y validación	67
3.6.4	Desarrollo de la red neuronal convolucional	67
3.6.4.1	RNC elaborada	68
3.6.5	Entrenamiento mediante una RNC	71
3.6.6	Instalación del S.O en el ordenador de placa	72
3.6.7	Instalación de versión de python superiores	73
3.6.8	Instalación de un ide y creación de entorno virtual	75
3.6.9	Testeo y predicción en tiempo real	76
3.7	Implementación del prototipo	79
3.8	Escenarios de pruebas	80
3.8.1	Escenario idóneo	81
3.8.2	Escenario con poca iluminación	82

N°	Descripción	Pág.
3.8.3	Escenario con mucho ruido	83
3.8.4	Escenario usuario se encuentra lejos	85
3.8.5	Escenario donde el usuario no cuenta con su extremidad completa	86
3.8.6	Escenario donde el usuario usa un guante	87
	Conclusiones	88
	Recomendaciones	89
	Anexos	90
	Bibliografía	113

Índice de Tablas

Nº	Descripción	Pág.
1	Características de una Red Neuronal Artificial	20
2	Proyectos más populares que ofrece Mediapipe	37
3	Especificaciones técnicas de Nvidia Jetson Nano Developer Kit	39
4	Comparaciones de lenguajes de programación para visión artificial	49
5	Comparaciones de microordenadores	51
6	Precio y Cantidades de Hardware utilizado	52
7	Precio y Cantidades de Software utilizado	52
8	Datos para obtener la muestra en base a la población	54
9	Requerimiento funcional #1	62
10	Requerimiento funcional #2	63
11	Requerimiento funcional #3	63
12	Requerimiento funcional #4	64
13	Requerimiento no funcional	64
14	Entidades principales del sistema	67
15	Librerías y versiones específicas para el sistema	79

Índice de Figuras

Nº	Descripción	Pág.
1	Lengua de Signos Americana	13
2	Diagrama de los pasos seguidos en un sistema de reconocimiento facial	15
3	Manipulación de datos para reconocimiento de imágenes	17
4	Esquema de Neurona	20
5	Red Neuronal Unicapa	21
6	Problema Linealmente Separable	21
7	Red Neuronal de Múltiples Capas, Capa Oculta y Capa de Salida	22
8	Problema No Linealmente Separable	23
9	Red Neuronal Profunda. Elaborada por Nielsen	23
10	Ejemplo de Imagen después de pasar por un filtro pasa bajo o suavizado	26
11	Análisis de píxeles de una matriz 3x3	26
12	Atenuación leve en la etapa de convolución de una imagen	27
13	Proceso de convolución de una imagen para extraer características	27
14	Max-Pooling aplicado a una imagen previamente subdividida en regiones	28
15	Average Pooling 2x2 y Max Pooling 2x2	29
16	Proceso de Flattening de una imagen	29
17	Procedimiento de una imagen en una red neuronal convolucional	30
18	Logo del sistema operativo libre de tipo Linux.	31
19	Hola mundo en lenguaje C++.	32
20	Anaconda Navigator interfaz y posibles interpretes a tener en cuenta	33
21	Librerías específicas de un ambiente virtual en el intérprete Pycharm.	34
22	Imágenes de manos sintéticas renderizadas con los 21 puntos de interés	37
23	NVIDIA Jetson Nano Developer Kit	40
24	Raspberry Pi 4	41
25	Cámara Web con Micrófono Incorporado de resolución 720p	42
26	Módulo de cámara de Raspberry	42
27	Abecedario de ASL 2020	43
28	Cantidad de personas con discapacidad en la familia	57
29	Cantidad de personas con discapacidad auditiva en la familia	58
30	Frecuencia en sitios de gran afluencia	59
31	Formas de comunicación	60

N°	Descripción	Pág.
32	Conocimientos sobre IA/visión artificial	61
33	Conocimientos sobre IA y sus soluciones	62
34	Conocimiento sobre apps de traducción	63
35	Preferencia del prototipo en sitio de gran afluencia	63
36	Requisitos para el sistema.	64
37	Diagrama de caso de uso de reconocimiento de manos	68
38	Diagrama de caso de uso seguimiento de manos	69
39	Diagrama de caso de uso predicción de seña	69
40	Resultado de algoritmo de recolección de datos	71
41	Resultados de recolección de letra A	72
42	Creación de Alfabeto basado en ASL	72
43	Forma general de una RNC	74
44	Representación gráfica de arquitectura original AlexNet	76
45	Inicio de entrenamiento con precisión menor al 20%	77
46	Iteraciones finales con precisión por encima del 80%	77
47	Creación y visualización de los pesos y modelos creados por la RNC	77
48	Versiones instaladas de Python en Jetson nano	80
49	Visual Studio Code en Jetson Nano	81
50	Activación de entorno virtual en el prototipo	82
51	Compilación del traductor de señas en placa Jetson Nano	83
52	Prototipo traductor de señas en funcionamiento	84
53	Prueba del traductor en escenario ideal	86
54	Prototipo funcionando en escenario ideal	87
55	Prueba del traductor de señas en escenario con poca iluminación	87
56	Prototipo funcionando en escenario con poca iluminación	88
57	Prueba del traductor en escenario con mucho ruido	89
58	Prototipo funcionando en escenario con poco ruido	89
59	Prueba del traductor en escenario en que el usuario se encuentra lejos	90
60	Prototipo funcionando en escenario con mucho ruido	91
61	Puntos de referencia de mediapipe	91
62	Código de recolección de data #1	97
63	Código de recolección de data #2	97

N°	Descripción	Pág.
64	Código de recolección de data #3	98
65	Librerías y funciones de la RNC	99
66	Parámetros establecidos para la RNC	99
67	Preprocesamiento de imágenes para mayor información	100
68	Inicio de la RNC y sus capas ocultas	101
69	Entrenamiento de la RNC y su creación de modelos	101
70	Creación de entorno virtual Entornodeprueba #1	102
71	Creación de entorno virtual Entornodeprueba #2	102
72	Activación y desactivación del entorno virtual	103
73	Librerías existentes en el prototipo	103
74	Prueba del traductor de seña a texto en lugar con gran afluencia #1	104
75	Prueba del traductor de seña a texto en lugar con gran afluencia #2	105
76	Prueba del traductor de seña a texto en lugar con gran afluencia #3	105
77	Prueba del traductor de seña a texto en lugar con gran afluencia #4	106
78	Prueba del traductor de seña a texto en lugar con gran afluencia #5	106
79	Inferencia en tiempo real del prototipo #1	107
80	Inferencia en tiempo real del prototipo #2	108
81	Inferencia en tiempo real del prototipo #3	108
82	Interfaz de “SD Card Formatter”	109
83	Interfaz de BalenaEtcher	110
84	Flasheo de JetPack SDK en SD	110
85	Inserción de SD con Jetpack SDK en Jetson Nano	111
86	Configuración inicial del S.O. en Jetson Nano	112
87	Configuración final del S.O. en Jetson Nano	112
88	Creación de usuario en JetPack SDK	113
89	Inicio de JetPack SDK en Jetson Nano	113
90	Interfaz de JetPack SDK	114
91	Jetson Nano developer kit y accesorios básicos a utilizar	115
92	Jetson Nano con periféricos conectados	115
93	Jetson Nano developer kit sin periféricos	116
94	Interfaz digital de JetPack SDK	116
95	Prototipo traductor de seña funcionando correctamente	117



**ANEXO XIII.- RESUMEN DEL
TRABAJO DE TITULACIÓN (ESPAÑOL)
FACULTAD INGENIERÍA INDUSTRIAL**



CARRERA INGENIERÍA EN TELEINFORMÁTICA

**“TRADUCTOR DE LENGUAJE DE SEÑAS BASADO EN VISIÓN ARTIFICIAL
PARA PERSONAS CON DISCAPACIDAD AUDITIVA.”**

Autor: Alarcón Monserrate Nelson Alexander

Tutor: Ing. Comp. Castillo León Rosa Elizabeth, Mg.

Resumen

El objetivo principal de esta investigación es realizar un prototipo de interpretación y traducción de gestos (únicamente letras) mediante visión artificial y Python. El desarrollo del prototipo empieza con la recolección de Data en la que se utilizaron aproximadamente 13000 imágenes para la creación de un Dataset propio. Lo siguiente fue el desarrollo de una red neuronal convolucional (RNC) que dará como resultado un modelo de entrenamiento basado en el Dataset previo. Posteriormente se traslada el sistema traductor de seña al prototipo Jetson Nano donde se ejecutará en un entorno virtual que contenga las librerías específicas para su funcionamiento. Finalmente, se muestran las pruebas realizadas en distintos escenarios donde se comprobó el funcionamiento del prototipo, notando que las inferencias alcanzaban un 80% de efectividad en pantalla. La importancia del prototipo radica en que mediante nuevas tecnologías y visión artificial se puede ayudar a toda una comunidad y fomentar la inclusión.

Palabras Claves: Visión Artificial, Discapacidad auditiva, Jetson Nano, Lengua de señas, Python, Red Neuronal Convolucional.



**ANEXO XIV.- RESUMEN DEL
TRABAJO DE TITULACIÓN (INGLÉS)
FACULTAD INGENIERÍA INDUSTRIAL**



CARRERA DE INGENIERÍA EN TELEINFORMÁTICA

“SIGN LANGUAGE TRANSLATOR BASED ON ARTIFICIAL VISION FOR PEOPLE WITH HEARING IMPAIRMENTS.”

Author: Alarcón Monserrate Nelson Alexander

Advisor: Ing. Comp. Castillo León Rosa Elizabeth, Mg.

Abstract

The main objective of this research is to make a prototype of interpretation and translation of gestures (only letters) using artificial vision and Python. The development of the prototype begins with the collection of data in which approximately 13,000 images were used to create its own dataset. The next step was the development of a convolutional neural network (CNN) that will result in a training model based on the previously created Dataset. Subsequently, the sign translator system is transferred to the Jetson Nano prototype where it will be executed in a virtual environment that contains the specific libraries for its correct operation. Finally, the tests carried out in different scenarios where the functioning of the prototype was verified, noting that the inferences reached 80% effectiveness on screen. The importance of the prototype lies in the fact that through new technologies and artificial vision an entire community can be helped and at the same time inclusion can be promoted.

Keywords: Artificial Vision, Hearing Impairment, Jetson Nano, Sign Language, Python, Convolutional Neural Network

Introducción

La lengua de señas consiste en una forma de comunicación que, mediante signos efectuados por las manos, las personas con discapacidad auditiva son capaces de poder expresarse de manera efectiva con su entorno social, se debe recalcar que es únicamente con otras personas con esta discapacidad o que conozca dicha lengua de señas específico empleado.

Ya que existe la creencia popular de que únicamente se cuenta con una sola lengua de señas que utilizan a nivel mundial las personas sordomudas, cuando la realidad es totalmente distinta y cada país cuenta con su propia lengua de signos, con el fin de representar tanto letras que no existen en otros abecedarios (Cómo en Ecuador con la Ch, Ll y Ñ), como palabras cuyo significado varía dependiendo del país. A pesar de que la mayoría son variaciones provenientes del ASL (American Sign Language).

Según la Federación Mundial de Sordos en el 2021 se pudo contrastar que existen aproximadamente 70 millones de personas con discapacidad auditiva a nivel mundial. Donde un poco más del 80% vive en países en desarrollo y se llega a utilizar más de 300 diferentes lenguas de señas (Algunos siendo variaciones de otros). (Voisard, 2015)

El último censo de población y vivienda de Ecuador señaló que la población se acerca a los 17,3 millones de personas y con una proyección para este próximo censo aplazado por pandemia superior a los 18 millones de habitantes. (INEC, 2021)

De este gran número de pobladores según El Consejo Nacional para la Igualdad de las Discapacidades (CONADIS), en Ecuador hay aproximadamente 66,538 personas con discapacidad auditiva registradas en donde el 54.49% corresponde a los hombres y el 45.51% a mujeres. (CONADIS, 2022)

Según INEC (Instituto Nacional de Estadísticas y Censos) afirma que dentro de Guayaquil la cual es considerada como la ciudad del comercio y la segunda más poblada en Ecuador con 2'723.665 habitantes. (El Universo, 2021)

Se puede encontrar alrededor de 9.693 personas con discapacidad auditiva registradas, de los cuales 4.388 pertenecen a mujeres, 5.301 correspondientes a hombres y 4 a personas de la comunidad LGBTI. (CONADIS, 2022)

El reducido entorno social de una persona con discapacidad auditiva en la ciudad de Guayaquil, la cual únicamente puede comunicarse a través de la lengua de señas con

personas que tengan conocimientos sobre el mismo, puede dificultar el realizar actividades diarias por tener que depender de terceros para comunicarse de manera efectiva y rápida como sucede en locales de comida rápida donde se cuenta con tiempos de espera bajos.

A pesar de la existencia de tecnología o alternativas que permita brindar un ligero soporte para este tipo de personas, sigue sin ser una solución del todo efectiva. Sin embargo, gracias a los avances constantes de la programación, inteligencia artificial y desarrollo de programas de lenguaje abierto cada vez más óptimos lo cual lleva a la siguiente pregunta central del trabajo ¿Cómo la Inteligencia/Visión Artificial puede facilitar a las personas con discapacidad auditiva actividades básicas como compras en lugares donde existan tiempos reducidos en la atención y que no se tiene conocimientos sobre la lengua de señas?

El objetivo de esta investigación es el de poder realizar un prototipo de un sistema de interpretación y traducción de gestos (Abecedario de ASL) mediante visión artificial y programación.

Para llevar a cabo esta investigación, el trabajo se ha estructurado en 3 capítulos. En el capítulo I se va a describir la problemática a solucionar, su justificación y objetivos. En el capítulo II va a encontrar lo referente al Marco teórico de la visión artificial y los dispositivos tecnológicos disponibles a utilizar en la creación del prototipo. En el capítulo III se va a relatar el tipo de metodología a emplear para la investigación, junto con los resultados que vaya a demostrar el prototipo cuando esté en funcionamiento.

Capítulo I

El Problema

1.1 Planteamiento del problema

Dentro de la ciudad de Guayaquil se puede encontrar alrededor de 9.693 personas con discapacidad auditiva registradas, de los cuales 4.388 pertenecen a mujeres, 5.301 correspondientes a hombres y 4 a la comunidad LGBTI. (CONADIS, 2022)

El lenguaje de señas es la principal forma de comunicación entre personas con este tipo de discapacidad sea auditiva o incluso vocal. No obstante, este tipo de personas tienen problemas de comunicación debido a que la mayoría en la sociedad no domina dicho lengua, viéndose afectados debido al nivel de dificultad que representa el ser comprendido, a pesar de que existen alternativas tecnológicas para poder ayudar a esta causa, son versiones básicas y abandonadas o debes tener equipado un dispositivo un poco molesto consigo mismo, por lo que suelen emplear la escritura como solución momentánea, ya que los intérpretes son muy escasos y de costo elevado. Teniendo como consecuencia la pérdida de independencia y privacidad, incluso existen personas sordomudas que no aprenden a escribir o de plano su escritura es muy poco práctica complicando más su situación social.

Las personas con discapacidad auditiva tienden a sentirse rechazados por sus dificultades de comunicación, por lo tanto, si les cuesta tanto tener una conversación con otras personas que no entienden la lengua de señas ¿Cómo haría un sordomudo para realizar compras de cosas necesarias como comida en un restaurante si el vendedor desconoce la lengua de señas?

Se conoce que dentro del país abundan franquicias de comida rápida en donde su mayor atractivo aparte de la comida, suele ser la atención y entrega de alimentos en el menor tiempo posible. Por ende, al ser muy frecuentados este tipo de restaurantes, es normal que un número reducido de su clientela suela pertenecer al porcentaje de personas con discapacidad auditiva. Debido al desconocimiento de la lengua de señas en el personal o la falta de tecnología inclusiva como pantallas en las que ordenas comida, es que suelen elevarse de manera descomunal los tiempos de atención y entrega respectivamente. Este tipo de situaciones es muy común observarla dentro de centros comerciales pues sus patios de comida albergan locales que brindan dicho servicio de comida rápida y al ser un espacio reducido, es evidente que se encuentren limitaciones

siempre sobre implementación de tecnología inclusiva o que permita solucionar este tipo de escenarios.

Por lo tanto, existe una situación en la que personas con una discapacidad sea de audición se sienten excluidas totalmente debido a que se les dificulta en muchas ocasiones realizar sus actividades diarias como conseguir alimentos debido a que por su falta de independencia se ven obligados a encontrar formas alternativas para poder tener comunicación como lo son la escritura, sin embargo, el problema crece cuando no son capaces de leer o escribir adecuadamente y solo tienen comprensión mediante la lengua de señas.

Abordar este problema traerá consigo beneficios prácticos en cuanto a tiempos de espera para restaurantes de comidas rápidas y una mejora en la atención al cliente para personas con una discapacidad auditiva. También se contribuirá a la comprensión de la lengua de señas dentro de la ciudad.

1.2 Formulación del problema de investigación.

¿Cómo la Inteligencia/Visión Artificial puede facilitar a las personas con discapacidad auditiva actividades básicas como compras en lugares donde existan tiempos reducidos en la atención y que no se tiene conocimientos sobre la lengua de señas?

1.3 Justificación e importancia.

Ante las escasas alternativas tecnológicas existentes dentro del país para poder brindar soporte de comunicación a la comunidad conformada por personas con discapacidad auditiva y la poca divulgación sobre la importancia que tiene la lengua de señas para estas personas.

La relevancia del tema radica en las aproximadamente diez mil personas con discapacidad auditiva registradas en la ciudad de Guayaquil y los problemas que presentan al momento de comunicarse con personas que no cuentan con los conocimientos sobre la lengua de señas que emplean.

La investigación surge de la necesidad por parte de las personas con discapacidad auditiva para poder comunicarse de manera eficiente en un entorno social donde se precise de respuestas rápidas debido a tiempos de reacción bajos como los de un restaurante de comida rápida y que a su vez no pierdan su independencia por terceros para poder ser comprendidos. Con el propósito de desarrollar un prototipo mediante visión artificial que pueda facilitar a través de inferencias en tiempo real dicha comunicación entre personas sordomudas y

aquellas que no cuentan con conocimientos sobre el ASL (American Sign Language) para romper así ese aislamiento social impuesto por el desconocimiento.

Este trabajo busca proporcionar datos útiles a una comunidad de universitarios sobre la visión artificial y los proyectos que se pueden llevar a cabo para poder beneficiar a una comunidad afectada. De manera que se pueda otorgar cierta independencia a las personas que padezcan de esta discapacidad, tener una inclusión por parte de negocios de comida rápida con su clientela y divulgar más sobre la lengua de señas.

Debido a que la Inteligencia artificial no es un tema con suficiente alcance en las universidades locales, el presente trabajo contribuye al aportar datos sobre la visión artificial para contrastarlos con otros estudios similares de origen local y extranjero, explicando las variantes al crear modelos de entrenamientos basado Redes Neuronales Convolucionales que pueden servir de base para futuros proyectos que vayan a utilizar componentes similares.

El trabajo tiene también una utilidad metodológica ya que podrán realizarse futuras investigaciones que hagan uso de metodologías compatibles, con el fin de lograr mayor cantidad de análisis, comparaciones y optimizaciones hablando netamente de los modelos. La investigación resulta factible porque se tienen disponible los recursos necesarios para lograrlo.

1.4 Objetivos de la investigación

1.4.1 Objetivo General

Realizar un prototipo de un sistema de interpretación y traducción de gestos (Abecedario de ASL) mediante visión artificial y Python.

1.4.2 Objetivos Específicos

- a. Recopilar información teórica referente para el estudio del estado del arte.
- b. Crear un repositorio básico de gestos basado en la detección de manos.
- c. Diseñar el sistema traductor de lengua de señas.
- d. Evaluar el funcionamiento del prototipo.

1.5 Alcance del proyecto

1.5.1 Delimitación Física

La ubicación geográfica donde se planea desarrollar este trabajo de investigación es en el centro comercial CityMall ubicado en el norte de la ciudad de Guayaquil.

1.5.2 Alcance

Diseño de un sistema basado en visión artificial y microordenador que permita traducción del abecedario de lengua de señas americano para poder agilizar el proceso de atención al cliente en franquicias de comida con una gran afluencia cuando una persona con discapacidad auditiva se acerque a los mismos, promoviendo también la inclusión entre consumidores.

Para el desarrollo del proyecto se tiene pensado seguir el siguiente proceso: Realizar una investigación previa sobre distintas herramientas de programación, aplicación o dispositivos tecnológicos existentes enfocados en inteligencia y visión artificial, que además estén disponibles en el mercado para utilizar.

Posterior a ello realizar un levantamiento de información con ayuda de clientes y personal de atención al cliente de cadenas de comida rápida que tengan mayores visitas. Para poder contrastar la necesidad existente de un sistema que permita reducción en sus tiempos de manera eficiente.

Finalmente, comprobar el funcionamiento del prototipo previamente diseñado y que mediante la programación crear modelos de entrenamiento óptimos para tener mayor precisión en la visión artificial que brinda.

En resumen, el alcance de este proyecto busca lograr con un set de datos de aprendizaje creado previamente distinguir las letras del abecedario American Sign Language mediante un prototipo que hará uso de programación y visión artificial para llevar a cabo el objetivo. Limitando a que se aplica a personas que cuenten con su extremidad completa, de tamaño promedio y en un ambiente controlado con iluminación y sin ruido.

Capítulo II

Marco Teórico

2.1 Antecedentes del Problema

Según (Chiguano, Moreno y Corrales, 2012) quienes realizaron el trabajo de titulación “Diseño e implementación de un sistema traductor de lenguaje de señas de manos a un lenguaje de texto mediante visión artificial en un ambiente controlado”, el cual estuvo enfocado en realizar un sistema que permita traducir el lenguaje de señas a lenguaje de texto con ayuda de la rama de la IA la visión artificial, de esta manera permitir que personas con una discapacidad auditiva que manejen el lenguaje de señas puedan comunicarse de manera efectiva con personas que no manejen este lenguaje. El proyecto fue realizado implementando técnicas de visión artificial diseñadas en el software Labview 2009 en conjunto con su respectiva toolkit de visión artificial (Cómo analogía vendría a ser una librería de visión artificial como OpenCV y Tensorflow para Python), una cámara web y un procesamiento digital de imágenes optimo conformado por filtros y operaciones morfológicas que permitían resaltar características de la imagen y eliminar información innecesaria que ralentice el mismo como lo es el ruido. Finalmente se asigna una clase a cada kinema y de esta manera al comparar en el sistema diseñado se puede visualizar en forma de texto la letra escrita en pantalla o a su vez se puede enviar a un documento de Word si lo que se requiere es escribir un párrafo. Las limitantes notorias en este proyecto es que al ser un poco antiguo el ambiente tenía que estar totalmente controlado con una excelente iluminación y fondo monocromáticos, ya que de lo contrario su eficiencia y acierto disminuirían considerablemente.

(Sivisapa Aguilera, 2016) indica en su trabajo de titulación llamado “Visión artificial aplicada para el reconocimiento del lenguaje de señas” como mediante lenguaje de programación C++ y algoritmos para el seguimiento de objetos segmentados a través del color para su posterior predicción, implementó algoritmos muy eficientes para ese año los cuales eran KNN (K-Nearest Neighbors) o conocido como K-Vecinos más cercanos y SVM (Support Vector Machine) ya que antes de perfeccionar y que se popularizaran el desarrollo de las neuroredes para visión artificial, se usaban con bastante frecuencia en trabajos similares que implicaban visión artificial con el fin de que la combinación de este tipo de algoritmos puedan reducir el porcentaje de error y acertar mucho más en las predicciones. El diseño de esta aplicación se realizó en un sistema operativo Linux para poder hacer que su distribución sea mucho más sencilla por ser un software libre, adicionalmente que

también no requiere de software o componentes de alta gama y por consiguiente caros para poder ejecutar dicha aplicación, sin embargo, en los resultados se observan que a pesar de compilar y tener un excelente porcentaje de aciertos, es indispensable estar en un ambiente controlado y con mucha iluminación como en el trabajo anterior debido a que es una condición para el reconocimiento en tiempo real de las señas realizadas por el usuario, cualquier variante de iluminación o fondos daba como resultado un ruido en la imagen que hacía imposible la correcta detección y etiquetado.

Según (Fernández Suárez, 2017) en su trabajo de titulación “Diseño y construcción de un prototipo de sistema electrónico para conversión de lenguaje de señas a mensajes de voz para la comunicación de personas sordomudas, en la ciudad de Chiclayo” afirma como para el procesamiento eficaz de imágenes utilizó una red neuronal artificial de tipo Retro propagación y Python debido a la naturaleza mismo del proyecto donde mediante un sistema electrónico se haría una posterior conversión de lenguaje de señas a mensajes de voz todo gracias a la combinación del diseño del software para la recepción de escritura y conversión de audio y un guante con sensores flex controlados por Arduino para la predicción de palabras. A partir de este punto se puede observar como el avance de las versiones de Python fueron abriendo paso para ser implementadas en proyecto de visión artificial debido a su versatilidad como lenguaje de programación y su repertorio de librerías, adicional que se incluían redes neuronales artificiales para el entrenamiento tenga buenos resultados en su predicción.

Respecto a (Aquino Castro, 2018) quien realizó el trabajo similar pero más actualizado de titulación “Reconocimiento e interpretación del alfabeto dactilológico de la lengua de señas mediante tecnología móvil y redes neuronales artificiales”, donde a diferencia del trabajo anterior aquí se implementaron redes neuronales artificiales que son un paso agigantado y más óptimas en el campo de la visión artificial. Adicional a ello se implementaron las librerías por defecto que funcionan con las redes neuronales como lo son OpenCV y Tensorflow, algo a destacar es que a diferencia de otros trabajos en los que se realiza una red totalmente desde cero y que se la entrena hasta que alcance porcentajes de aciertos adecuados, aquí lo que se hizo es el método de “Transferencia de aprendizaje” que consta de reentrenar una red neuronal convolucional que ya ha sido previamente entrenada con el fin de optimizar el modelo resultante de la red. Para finalmente obtener un sistema que reconozca de manera acertada el abecedario dactilológico boliviano y este mismo sistema pasaría a ser implementado en una aplicación para únicamente dispositivos móviles

que cuenten con un sistema operativo Android ya que son más permisivos al momento de otorgar permisos para instalación de aplicaciones externas a la playstore, además de que la mayoría de usuarios de móviles utilizan este sistema operativo Android por lo que se podría llegar a mucha más gente. Los resultados obtenidos en este trabajo eran satisfactorios siempre y cuando se cumpliera la condición de estar en un ambiente controlado, a cierta distancia y muy importante la iluminación adecuada para así evitar el exceso de ruido en las imágenes.

Según (Mancilla, Vásquez, Arguijo y cols, 2019) relatan en su publicación “Traducción del lenguaje de señas usando visión por computadora” el desempeño que pueden tener las redes neuronales de tipo multicapa y un clasificador para el reconocimiento de señas del LSM (Lengua de Señas Mexicana) en donde se limitan a 21 kinemas estáticos y se omiten los dinámicos, ya que hasta el momento solo se les dan avances a los abecedarios dactilológicos indio y japonés por lo que optaron por diseñar una red neuronal convolucional considerando tres conjuntos de características para notar el incremento de eficiencia en los aciertos. Estos conjuntos de características son: Momentos de Hu, Momentos de Zernike e Histogramas de orientación del Gradiente (HOG), donde se demostró en los resultados que lo que arrojó mayor eficiencia en la predicción en tiempo real fue momentos de Zernike. Destacando que este fue un complemento luego de la técnica de preprocesamiento de imágenes en las que se puede encontrar las etapas de: La reducción de ruido, escalamiento de la imagen y eliminación de fondo de la imagen para su posterior extracción de conjunto de características más relevantes.

Según Estévez (2022) en su trabajo de titulación “Desarrollo e implementación de un sistema de identificación de signos de la lengua de señas mexicana basado en redes neuronales y el sistema embebido Jetson Nano” como mediante Google Colaboratory que es una excelente herramienta online para programar y diseñar proyectos con fines de Inteligencia artificial, creo una red neuronal convolucional que al variar parámetros como neuronas, números de filtros o capas de convolución tiene como resultado varios modelos para ponerlos a pruebas con las inferencias en tiempo real y ver cual arroja valores óptimos, para posterior a ello mediante algoritmos darle un formato compatible con el micro ordenador para hacer un tipo de proyecto semi portable pues aún requiere de una fuente de alimentación constante para poder funcionar. Para el proyecto fue imprescindible una base de datos recolectada muy grande para tener mayor porcentaje de acierto y se inspiró en el lenguaje de señas colombiano, americano y bengalí conteniendo en su data set final más de

10k de imágenes de entrenamiento. Las limitaciones nuevamente se ven reflejada en la exclusión de letras dinámicas como lo son la J y Z y un ambiente controlado, pero mucho menos exigente que en proyectos pasados.

Como se ha relatado a lo largo de los antecedentes, se puede observar como a inicios hasta mediados de la década pasada se implementaban algoritmos populares o incluso filtros acompañados de herramientas o librerías visuales para poder mediante un software predecir caracteres pertenecientes al abecedario dactilológico de origen. Debido a la época se puede observar que era necesario muchos procedimientos adicionales para poder hacer una predicción en el menor tiempo posible.

A finales de la década pasada se puede notar como mediante Python e incluso C++ es posible la creación e implementación de redes neuronales convolucionales que significaron un avance significativo en proyectos de visión artificial, ya que una vez creada, el entrenamiento se basaba en parámetros que el usuario podía alterar para poder tener mayor eficacia en la predicción y acierto en el modelo resultante de la red. Finalmente existe la posibilidad de incluir todo en un microordenador para no necesariamente ocupar equipo de alta gama y potente sin sacrificar eficacia en las predicciones.

2.2 Fundamentación teórica

2.2.1. Personas con discapacidad auditiva

La discapacidad auditiva es “La deficiencia de las funciones sensoriales relacionadas con la percepción de los sonidos y la discriminación de su localización, tono, volumen y calidad”. (OMS, 2001, p.68)

Por ende, las personas que padecen de esta discapacidad se encuentran en constante dificultad para lograr comunicarse, esto tiene una considerable consecuencia en el desarrollo cognitivo del individuo, así como emocional y social debido a que no existe una inclusión en varios entornos, sino lo contrario.

Si bien es cierto, hay alternativas tecnológicas-médicas que permiten a la persona con una discapacidad auditiva el recuperar el habla (Siempre y cuando sus cuerdas vocales no se vean afectadas) y en un pequeño porcentaje su audición gracias a implantes cocleares y un aprendizaje intensivo por lo que se recomienda realizar este proceso quirúrgico cuando son infantes para mayor eficiencia.

Sin embargo, al ser un proceso extremadamente caro hace que no sea posible para la mayor parte de la población con este padecimiento poder acceder a estos implantes. Por lo que su alternativa y la más utilizada a nivel mundial para poder expresarse de manera eficiente sin que su desarrollo cognitivo se haya visto afectado durante su infancia, es por lengua de signos.

2.2.2. Lengua de señas

Existe una gran controversia cuando se refiere a la lengua de señas por lenguaje de señas, debido a que ambas palabras tienen significados distintos.

Según la D.R.A.E “lengua es un sistema lingüístico que se caracteriza por estar plenamente definido, por poseer un alto grado de nivelación, por ser vehículo de una cultura diferenciada y, en ocasiones, por haberse impuesto a otros sistemas lingüísticos”. (Storch de García, 2020)

Por lo tanto, se puede entender por lengua es ese código que se puede aprender y se utiliza para poder comunicarse entre dos o más personas. Recalcando que la lengua viene a ser considerada un idioma que tiene su estructura, gramática, vocabulario y gestos totalmente propios.

Retomando lo dicho por Storch de García (2020) “El lenguaje es un sistema de comunicación y expresión verbal propio de un pueblo o nación”. Lo que se puede interpretar como la capacidad que tiene cada ser humano para comunicarse con otro individuo a través del sonido que emana oralmente. Por lo que el término correcto en el idioma español para referirse a la comunicación mediante gestos es “Lengua de señas” en algunas partes como España, suele ser “Lengua de signos”, sin embargo, en Latinoamérica suele emplearse el término “señas” debido a que algunos consideran muy redundante el tener Lengua y signos en la misma oración.



Figura 1. Lengua de Signos Americana. Adaptada de www.clikisalud.net/. Elaborado por Fundación Carlos Slim.

2.2.3. Inteligencia Artificial

Oracle menciona que “La Inteligencia Artificial, también denominada IA se refiere a sistemas o máquinas que imitan la inteligencia humana para realizar tareas y pueden mejorar iterativamente a partir de la información que recopilan”. (Oracle, 2020)

En la actualidad existe una gran variedad de usos o en proyectos en los que se puede implementar la IA. Los casos más populares por ser de gran utilidad, comerciales o novedosos son:

- Chatbots que utilicen IA: Pues mediante entrenamiento previo, suelen comprender los problemas más frecuentes de los clientes y proporcionar respuestas más eficaces.
- Los asistentes inteligentes como lo son los más populares del mercado: Alexa, Siri, etc... Pues tienen implementada una IA que les permita analizar información crítica que proviene de una Big Data o conjuntos muy grandes de datos para realizar una actividad de manera óptima. Actualmente se pueden notar avances muy importantes en la programación, donde mediante asistentes inteligentes suelen recomendar el resto del código de programación dependiendo de las librerías o código escrito con el fin de ser más rápidos y evitar enfermedades como túnel carpiano en los desarrolladores.
- Motores de recomendación: plataformas de streaming proporcionan recomendaciones automatizadas en base a los hábitos constantes que tiene el usuario final.

En resumen, la IA ayuda a poder reconocer patrones de comportamiento, resolver mediante análisis problemas frecuentes, reconocimiento de imágenes (Mediante visión artificial que es una rama directa de la IA) y aprender a realizar tareas nuevas mediante un previo entrenamiento y aprendizaje automático.

2.2.4. Visión Artificial

“La visión artificial es una disciplina científica formada por un conjunto de técnicas que permiten la captura, el procesamiento y el análisis de imágenes”. (Domínguez, 2021, p.1)

Es decir, la visión artificial tiene como objetivo principal que un ordenador sea capaz de imitar la visión humana de manera que pueda extraer información relevante de las imágenes mediante un entrenamiento previo para responder a incógnitas previamente planteadas.

La informática e incluso ingeniería han colaborado para en sus inicios tener una serie de técnicas necesarias que ayudan a conseguir este objetivo principal. Como es costumbre este tipo de técnicas son de naturaleza matemática y sumamente complejas, haciendo que sean únicamente comprensible para especialistas en la materia. Sin embargo, existen opciones que permiten que esta rama de la IA pueda ser utilizada y sacar un mejor provecho por personas que no sean especialistas en algoritmos complejos. Por ejemplo, contar con una librería llamada “OpenCV” que permite tener una gran variedad de funciones a nuestro alcance al momento de programar para poder conocer y aprovechar las bases de la visión artificial mediante cámaras externas.

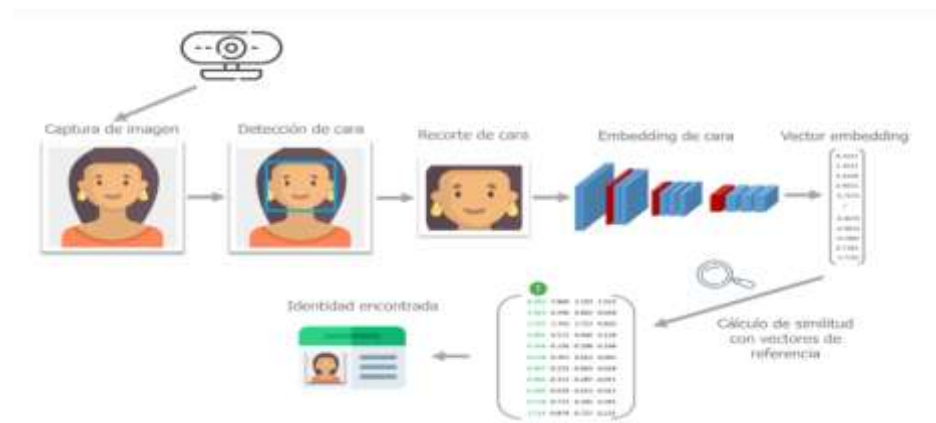


Figura 2. Diagrama de los pasos seguidos en un sistema de reconocimiento facial. Información obtenida de <https://www.cienciadedatos.net/documentos/py34-reconocimiento-facial-deeplearning-python.html> Elaborado por Joaquín Amat Rodrigo.

2.2.4.1 Campos de aplicación de visión artificial

Domínguez (2021, p.1) afirma que los beneficios y usos que tiene la visión artificial son numerosos. Pero destaca cuatro campos de suma importancia y que tendrán relevancia siempre al hablar de esta rama de la IA y son:

- Seguridad: Se encuentran los sistemas de reconocimiento facial, es común ver esto en empresas que resguarden información sensible o incluso es utilizada por los bancos para poder comprobar la identidad del usuario al realizar transacciones poco comunes.

- Industrial: Al momento de hacer comparaciones de imágenes de objetos manufacturados con las de otros usados previamente como modelos. De manera que al automatizar mediante la visión artificial el proceso de comparación de repuestos se podría automatizar también los procesos de control de calidad.

- Comercio: Dar a conocer al cliente las diversas oportunidades de como quedaría un producto final, logrando encaminar la compra. Normalmente las empresas que suelen sacar provecho de esto son las que están orientadas al negocio textil.

- Educación: Ya sea la creación de proyectos con propósitos metodológicos para gente que quiera especializarse en el área, como desarrollo de programas que ayuden a los infantes como un clasificador de figuras geométricas mediante cámara que permita la interacción con el niño en cuestión y este reciba retroalimentación para obtener beneficios cognitivos.

2.2.5. Machine Learning

El Machine Learning también denotado como ML, es el proceso mediante el cual se utilizan complejos modelos matemáticos de datos obtenidos previamente para ayudar a un equipo a aprender sin instrucciones directas. (Microsoft, 2022)

El ML o aprendizaje automático se considera un subconjunto de la inteligencia artificial. El ML utiliza algoritmos para la correcta identificación de patrones en los datos proporcionados, dichos patrones servirán para la creación de un modelo de datos capaz de realizar predicciones. Cabe recalcar que la precisión y porcentaje elevado de acierto es proporcional a la experiencia y datos proporcionados con anterioridad. Como analogía se puede decir que la práctica hace al maestro en humanos, de la misma forma en aprendizaje automatizado entre más datos y más entrenamientos, tendrá una mejor precisión.

El ML ha contado con numerosos proyectos exitosos en cuanto a problemas de clasificación se refieren. Cada uno de estos se diferenciaba del resto ya que no hacían uso de un algoritmo específico, sino que tenían su propio entrenamiento y datos a utilizar dependiendo de la naturaleza del mismo. Por ejemplo, se tiene la documentación de detección de signos escritos a mano realizada por Chen y cols. (2019), por mencionar uno de los más relevantes en los últimos años.

2.2.5.1. Técnicas de Machine Learning

Existen tres tipos de técnicas imprescindibles y consideradas principales que los usuarios utilizan en el machine learning o aprendizaje automatizado:

2.2.5.1.1. Aprendizaje supervisado

En este tipo de aprendizaje, es necesario tener una solución previamente establecida. Ya que se encarga de enseñar a un algoritmo como realizar eficientemente su trabajo cuando se cuenta con un conjunto de datos y así poder llegar a la solución esperada. (Rojas, 2018)

Es decir, se caracteriza por tener etiquetas puestas por nosotros mismos de posibles respuestas. Es muy común y el más sencillo de los ejemplos serían sobre el reconocimiento de imágenes, como por ejemplo enseñar al ordenador a reconocer entre un perro y un gato. Evidentemente se puede utilizar un repertorio muy extenso de imágenes de cada clase, pues cada opción se denomina clase y a su vez estos datos estarán clasificados de manera oportuna. Lo siguiente es realizar el entrenamiento del algoritmo y consigo tener un modelo preparado que me dará dos respuestas dependiendo de los datos que yo le vaya proporcionando. En este ejemplo ya se da a conocer las dos posibles respuestas pues únicamente son dos tipos de animales, pero para un ordenador es un proceso súper complejo que gracias a los algoritmos actuales puede en pocos pasos y con un poco de tiempo tener la habilidad de reconocer al igual que las personas, obviamente de manera artificial.

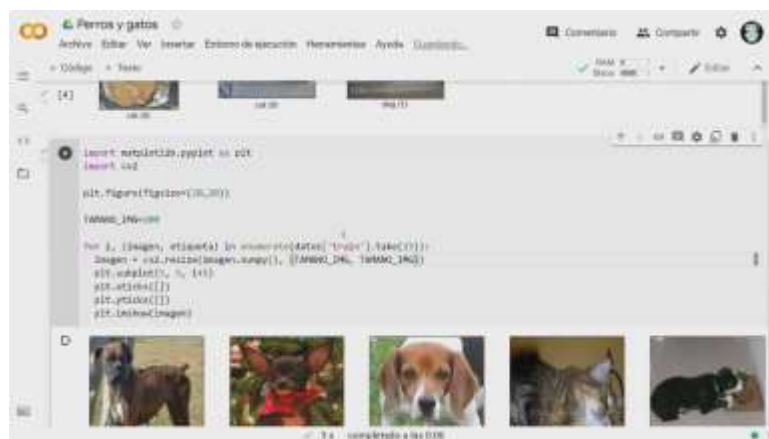


Figura 3. Manipulación de datos para reconocimiento de imágenes. Información adaptada de la web. Elaborada por Ringa Tech.

2.2.5.1.2. Aprendizaje no supervisado

El aprendizaje no supervisado es el segundo enfoque que puede tener el ML, y Rojas (2018) afirma. “Es cuando un algoritmo analiza el dato que no ha sido etiquetado con una respuesta para identificar patrones o correlaciones”.

Es decir, trata de un sistema capaz de analizar un gran conjunto de datos de un cliente y obtener resultados que indiquen lo que se desea buscar. Por ejemplo, los bancos suelen utilizar este tipo de aprendizaje para evitar el posible Churning (Abandono o suspensión de un servicio) de sus clientes. Para ellos se analizan grandes cantidades de datos que no necesariamente están etiquetadas, como por ejemplo los movimientos recientes en sus tarjetas, saldo en sus cuentas, entre otros. De manera que la combinación de dos o más de estos datos sin etiquetar permite deducir las elevadas posibilidades de que el cliente cese de nuestros servicios y que hacer para poder retenerlo.

2.2.5.1.3 Reforzar el aprendizaje

No es más que la implementación de un agente (Puede ser un programa informático que actúa en nombre de algo o alguien), con el fin de reemplazar al operador humano, esto ayuda a determinar el resultado final en función de un bucle de comentarios.

Este tipo de aprendizaje permite a una IA planear estrategias efectivas en base a la experimentación con los datos.

Consiste en una manera de optimizar en base a datos. Es decir, el ordenador puede tener la capacidad para aprender a partir de la experiencia propia, esto al momento de interactuar con el entorno en que se encuentre hasta dar con la forma más adecuada o ideal de tener.

Con la información disponible que tenga el ordenador, tomará acciones que volverá o no a repetir dependiendo de si en el refuerzo la recompensa es positiva o negativa. Por lo que al momento de comportarse de manera óptima es necesario otorgar una recompensa positiva para que pueda seguir mejorando, sin embargo, si es lo contrario es necesario una recompensa negativa para que pueda estar al tanto de que esa línea no debe seguirse.

Su funcionamiento más que tomar decisiones o hacer predicciones, el aprendizaje por refuerzo permite, entre otras aplicaciones, un mantenimiento predictivo o la personalización de experiencias de cliente.

2.2.6. Redes Neuronales Artificiales

Las redes neuronales artificiales o simplemente neuroredes, están creadas con el fin de poder modelar la forma de procesamiento de la información en sistemas nerviosos biológicos.

La importancia de las neuroredes consta que se consideran objeto de programación cuya intención es imitar el funcionamiento de las neuronas biológicas humanas. Estas neuronas se conforman por medio de una compleja interconexión de redes, a su vez las mismas redes cuentan con una organización que permite la posibilidad de interacción con el mundo y esto se explica con el fin de tener una analogía de lo que realiza un ordenador que se encuentra constituido por una variedad de componentes de procesamiento que también están interconectados, de manera que puedan procesar así la información en respuesta para algún tipo de recompensa o estímulo externo impuesto por el usuario a cargo.

(Izaurieta y Saavedra, 2000) afirman que el cerebro humano corresponde a un sistema altamente complejo, no lineal y paralelo, es decir, que realiza varias operaciones en simultaneo a diferencia de los ordenadores comunes que son de tipo secuencial.

En resumen, basado en estas definiciones se procede a conceptualizar que las neuroredes son procesadores de información y de una distribución paralela, que se constituye por muchas neuronas que vienen a ser cada una, una unidad sencilla de procesamiento y que al haber una gran variedad en conjunto forma una red capaz de procesar grandes cantidades de datos con el fin de actuar e imitar la naturaleza humana.

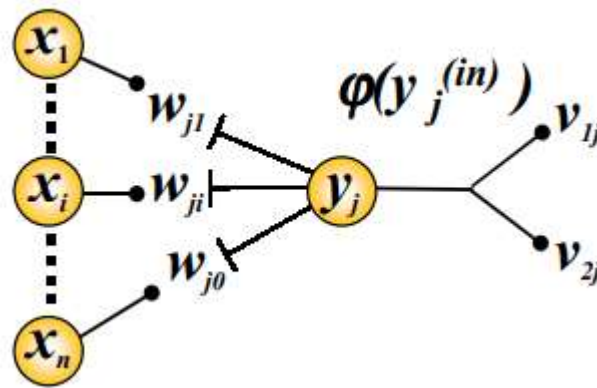


Figura 4. Esquema de Neurona. Elaborada por Fernando Izaurieta y Carlos Saavedra (2000)

En la siguiente tabla se describen las 3 principales características que (Gestal,2014) considera debe tener una neurored:

Tabla 1. Características de una Red Neuronal Artificial.

Topología de red:	Determina como una cantidad x de neuronas se encuentran distribuidas en capas y también entre sí. Dependiendo del problema a resolver, la topología de red será distinta pues se debe llegar a una óptima.
Regla de Aprendizaje:	La neurored es la integración de múltiples métodos de aprendizaje, de tal manera que tendrán la capacidad de poder aprender y dependiendo de ello elevar un porcentaje de acierto todo gracias a un entrenamiento previo.
Tipo de entrenamiento:	Una neurored consta normalmente de dos formas de entrenamiento. La primera es que en la etapa de aprendizaje la red se entrena para que los pesos obtenidos se adecuen a la red. La segunda es una etapa de ejecución donde la red toma un valor de funcionamiento real para poder tornarse totalmente operativa.

Fuente: Basado en Gestal (2014)

2.2.6.1. Tipos de Redes Neuronales Artificiales

Según (Estévez, 2022) indica que existen tres tipos principales de neuroredes y son:

2.2.6.1.1. Redes Neuronales Artificiales de Una Sola Capa

El tipo Unicapa en una neurored viene a ser la forma más simple de una red neuronal artificial la cual consiste únicamente en ir hacia adelante.

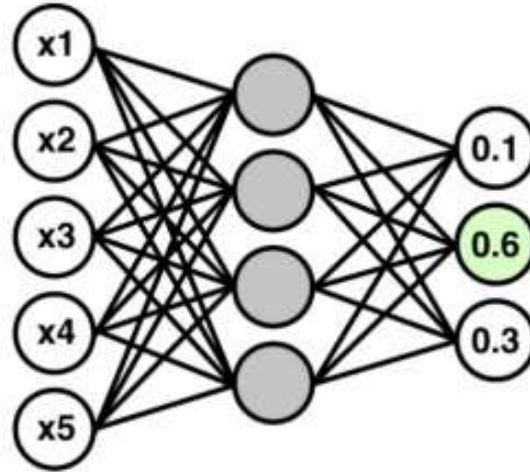


Figura 5. Red Neuronal Unicapa. Elaborada por Wong (2017)

La unicapa es el tipo de red que únicamente podrá resolver problemas linealmente separables. En este caso la única capa con la que cuenta es la que hace referencia a la capa de salida de todas las neuronas, en este caso no se puede contar la capa de entrada ya que en la misma solo se proporcionan datos y no existen cálculo de pesos.

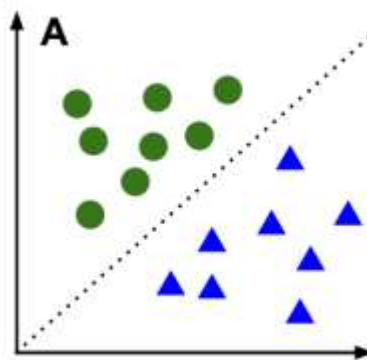


Figura 6. Problema Linealmente Separable. Elaborada por Palomera (2015).

2.2.6.1.2. Redes Neuronales Artificiales de Múltiples capas

Este tipo de neurored del tipo hacia adelante como las de una sola capa, se diferencian de las anteriores por contar con una o más capas adicionales de naturaleza oculta que terminarán realizando algún cálculo que influye en la salida de la red. Por capas ocultas se refiere a que

las mismas no son visibles directamente desde la capa principal de entrada o la final de salida.

Este tipo de capas se encuentran ordenadas por la forma en que reciben la señal desde la parte de entrada hasta la parte final de salida, denominando así a este tipo más complejo de conexiones como “hacia adelante”. (Ballesteros, 2015)

Las neuroredes de multiples capas son las más comunes de observar en los proyectos que requieran inteligencia artificial por su versatilidad a la hora de entrenar un modelo o adaptar uno previamente encontrado desde la red. Son más completos pues ofrece una ventaja que es la de multiples opciones y respuestas a lo largo de su ejecución y de poder mejorar la precisión de los proyectos considerablemente debido a la data otorgada.

Como se puede observar en la Figura 7, se puede notar como incluso se cuentan con más capas de salida y múltiples datos de entrada que posterior a ello entrarán para su depuración en las numerosas capas ocultas.

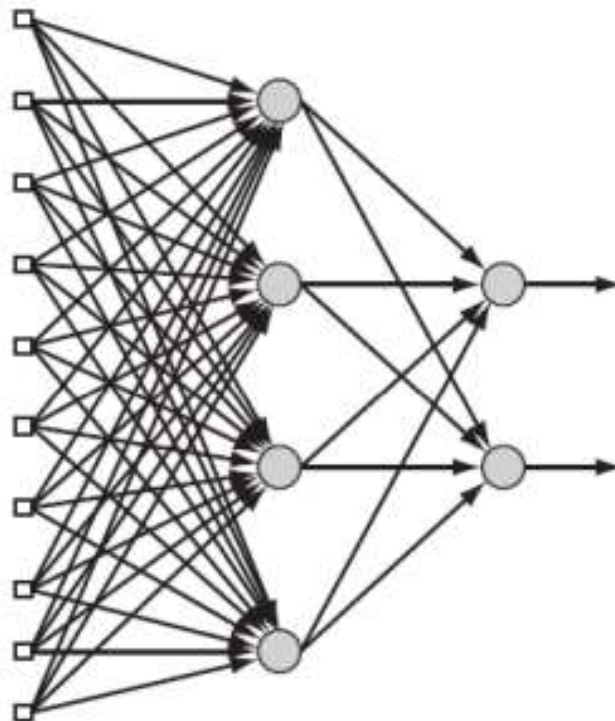


Figura 7. "Red Neuronal de Múltiples Capas, Capa Oculta y Capa de Salida. Elaborada por Haykin y cols. (2009).

A diferencia del primer tipo de neurored donde su uso es principalmente para operaciones lineales, esta de aquí puede resolver problemas de origen no lineales tales como patrones, predicciones y aproximaciones, entre otras.

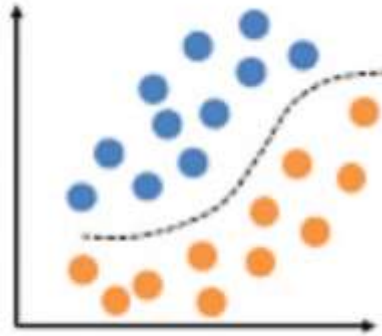


Figura 8. Problema No Linealmente Separable. Elaborada por Palomera (2015).

2.2.6.1.3 Redes Neuronales Artificiales Profundas

Una neurored profunda es cuando una red neuronal artificial cuenta con múltiples capas ocultas entre sus capas de entrada y de salida, las cuales son capaces de modelar la relación no lineal compleja. (Nielsen, 2015)

Debido a que existen varios niveles dentro de esta neurored, se puede enseñar al ordenador representaciones de los datos con distintos niveles de abstracción.

En la figura 9 se observa una neurored que cuenta con una capa de entrada múltiple, aproximadamente tres capas ocultas y una capa de salida. Esta imagen sirve de referencia para poder presenciar el cambio respecto a las predecesoras, ya que es común que en IA se utilice capas profundas con mayor número de capas ocultas.

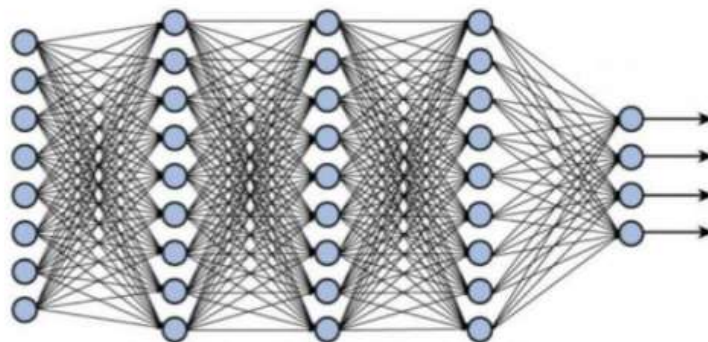


Figura 9. Red Neuronal Profunda. Elaborada por Nielsen (2015)

2.2.7. Redes Neuronales Convolucionales

Massiris, Delrieux y Fernandez (2018) comentan qué el aprendizaje profundo o Deep Learning (DL) permite que modelos computacionales compuestos por varias capas de

procesamiento puedan aprender representaciones sobre datos con múltiples niveles de abstracción.

Gracias a este concepto se puede interpretar que el DL es la representación muy precisa de manera automatizada sobre grandes volúmenes de datos previamente recolectados. En la actualidad el DL ha tenido una mejora notoria en cuanto a visión artificial gracias al reconocimiento preciso y acertado de imágenes y videos.

Un tipo de DL vienen a ser las redes neuronales convolucionales que actualmente constituyen el estado del arte en varios problemas de visión computacional. (Massiris y cols, 2018)

Su gran capacidad para actuar de manera acertada en interpretación y reconocimiento de imágenes y videos es debido a sus características fundamentales como lo son: conexiones locales, pesos, pooling y la implementación de una red profunda de multicapa.

(Estévez, 2022) sostiene que la red neuronal convolucional se divide en dos partes, la etapa convolucional y la de clasificación. Donde la primera ayuda a extraer características propias de cada imagen para posterior a ello pasarlo por distintos filtros a cada imagen, proceso el cual se denota como Convolución.

Normalmente esta etapa suele complementarse con algún tipo de submuestreo con el fin de reducir costo y recursos computacionales al momento de aprendizaje en la red neuronal convolucional debido a las grandes cantidades de características que se obtienen en ese proceso. Para evitar esto se usa la técnica de aplanamiento de datos antes de insertar los datos en la etapa de clasificación. Lo cual lo que hace es aplanar una imagen de 3 dimensiones a 2 dimensiones para hacerla más fácil de analizar.

La etapa de clasificación no es más que algún tipo de red neuronal del tipo multicapa tradicional en donde el número total de neuronas de la capa de salida por obligación debe coincidir con el número de clases que se tiene en la red. (Estévez, 2022)

2.2.7.1. Etapa Convolucional de las Redes Neuronales Convolucionales

La convolución es una operación matemática que se realiza sobre otras dos funciones con el fin de tener como resultado una tercera que se suele interpretar como una versión filtrada o mejorada de una de las dos primeras funciones originales. (Berzal, 2019)

Retomando a Estévez (2022) comenta que la convolución de las matrices I de dimensiones n , m y k de dimensiones a , b que se suele denotar mediante el símbolo de asterisco se representa de la siguiente manera:

$$[I * k](i, j) = S(i, j) = \sum_n \sum_m I(n, m) \cdot k(i - n, j - m)$$

Ecuación 1. *Fórmula matemática de convolucion de matrices. Elaborada por Estévez (2022)*

Donde la S viene a ser la matriz de salida, la I viene a ser la matriz de entrada y la k es el kernel (pixel principal de una matriz de pixeles) de convolución.

Lo que la ecuación trata de demostrar es la inversión de un kernel de convolución l y el desplazamiento que tendrá a través de los elementos de la matriz de entrada denotado por la letra I a la vez que se realiza la multiplicación de tipo punto de los elementos.

En la figura 10 se observa como los filtros que se aplican a la imagen dan como resultados variaciones de la misma, ya que el usuario puede elegir atenuarla y quitarle calidad mediante un filtro de suavizado o conocido como filtro pasa bajos, cambiarlo a escala de grises para un procesamiento más sencillo o incluso agregar ruido para una segmentación de tez de piel de interés.

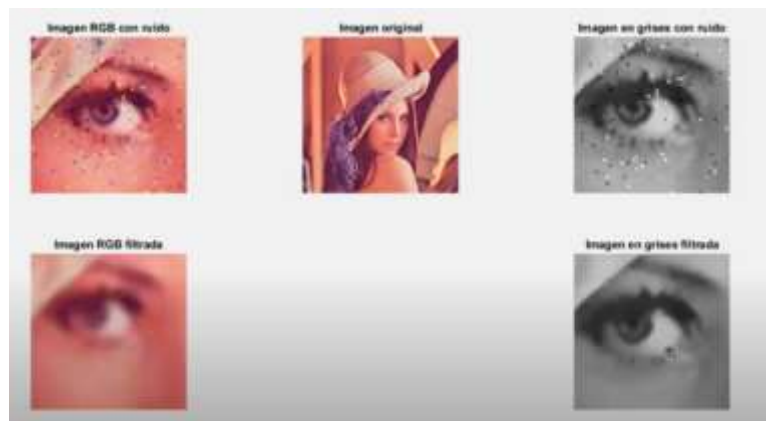


Figura 10. "Ejemplo de Imagen después de pasar por un filtro pasa bajo o suavizado". Elaborada por Mundo Tecnológico (2020).

En base a lo visto hasta ahora se puede decir que en una etapa de convolución para reconocimiento de imágenes se quita calidad a la imagen de entrada para que su análisis de pixeles sea mucha más rápida y eficiente. Sabiendo que los píxeles van desde el 0 siendo una tonalidad totalmente opaca (negra) hasta el 255 una tonalidad totalmente clara (blanca).

El procedimiento es el siguiente y como se ha demostrado la matriz más común a utilizar es una de 3×3 pues el analizar un solo pixel a la vez es muy ineficiente, de manera que, al analizar una matriz de pixeles se puede notar si existen cambios drásticos respecto al pixel

de interés que siempre se va a situar en el centro de la matriz. De esta forma comenzar a crear una especie de patrones que ayudarán en el posterior entrenamiento.



Figura 11. Análisis de píxeles de una matriz 3x3. Adaptada de Ringa Tech (2021)

Hay que recalcar que en este proceso es común que la imagen de salida perteneciente a S se le denomine como versión filtrada o modificada de la imagen y sus características. También es necesario comprender que no solo se atenúan características en las imágenes de entrada, sino que existe la posibilidad también de poder acentuar características de las imágenes de entrada (Mejorar curva, colores o incluso bordes). Esto se puede demostrar con las figuras 12 y figura 13 respectivamente.

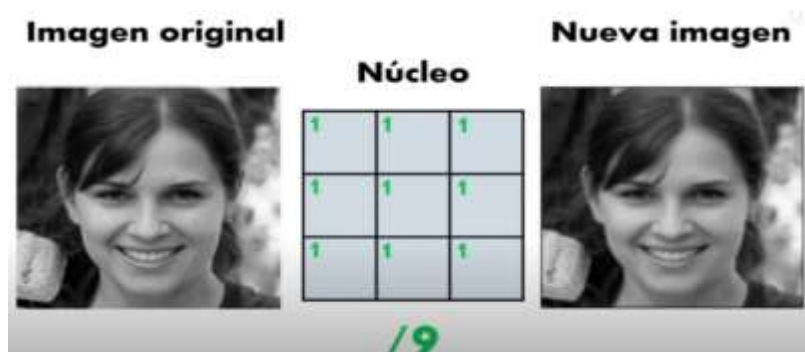


Figura 12. Atenuación leve mediante el kernel de interés en la etapa de convolución de una imagen. Adaptada de Ringa Tech (2021).



Figura 13. Proceso de convolución de una imagen para extraer características más importantes para su posterior agrupación máxima. Adaptada de Ringa Tech (2021).

Por lo que se puede deducir que los filtros convolucionales son una serie de pesos (rango de números definidos al azar) que se aplican a los valores de píxel en cada imagen de entrada. Como es un constante entrenamiento se espera que los pesos logren aprendizaje automático y mejoren su precisión.

2.2.7.1.1. Capa de Pooling

El objetivo principal de la capa de Pooling no es más que acumular una gran cantidad de características de los mapas generados cuando una imagen pasa por un filtro de convolución.

Normalmente lo que se busca en esta función es la reducción constante del tamaño de la imagen con el fin de reducir también la cantidad de parámetros y ahorrarle trabajo al ordenador, así como prevenir un sobre ajuste de la red neuronal artificial (Cárdenas, 2020)

Retomando a Granda (2020) indica que la forma más común que se puede encontrar del Pooling es “Max-Pooling”, el cual no hace más que aplica un filtro “Max” a una subregión de la imagen en donde ya se obtuvieron la matriz de píxeles y en base a ello obtener el píxel con el máximo valor de la región. Para comprender mejor se puede notar en el ejemplo de la *figura 14*.



Figura 14. Max-Pooling aplicado a una imagen previamente subdividida en regiones. Elaborada por Granda Cárdenas (2020)

Existen variaciones de pooling como lo son el average pooling en el cual a diferencia del max pooling aquí se busca el valor medio de cada subconjunto de la matriz como se muestra las diferencias en la **figura 15**.

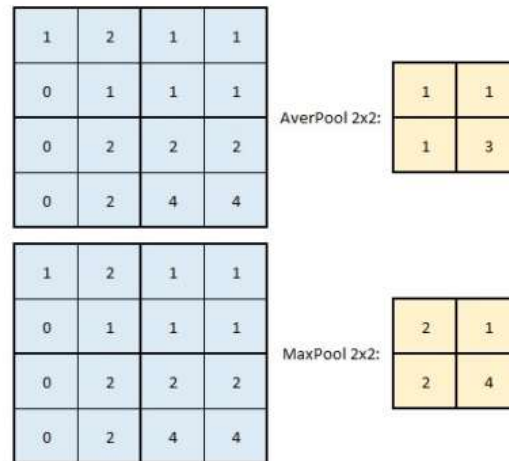
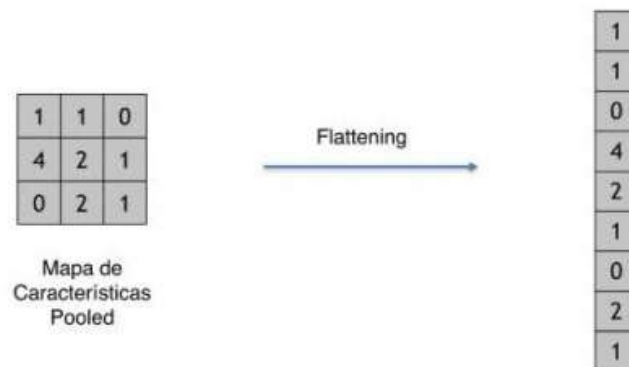


Figura 15. Average Pooling 2x2 y Max Pooling 2x2. Elaborado por Méndez (2019)

2.2.7.1.2. Capa de Flattening

El aplanamiento o mejor conocido como flattening se realiza con los resultados de la imagen una vez pasó por los filtros de convolución y hayan reducido una dimensión gracias a la capa de Max-Pooling que va a permitir acomodar el mapa de características en un vector



unidimensional, para posterior a ello ser los datos de la capa de entrada de la red neuronal artificial. Este procedimiento puede explicarse de manera más sencilla con la **Figura 16**.

Figura 16. Proceso de Flattening de una imagen. Elaborada por Gomilla (2019)

2.2.7.2. Etapa de Clasificación de las Redes Neuronales Convolucionales

Finalmente, mediante el método de aplanamiento que se explicó previamente para poder quitar características a imágenes restándoles una dimensión para un análisis más óptimo

hace que se obtenga un vector de datos unidimensional el cual se introduce a la capa de entrada de una red neuronal artificial de tipo multicapa para poder realizar la clasificación de la imagen de entrada.

Como se especificó en esta red convencional el número de neuronas debe como requisito indispensable ser igual al número de clases a identificar.

En la figura 17 propuesta por Gomilla (2019) se observa un resumen de los pasos a los que se somete una imagen en una red neuronal convolucional.

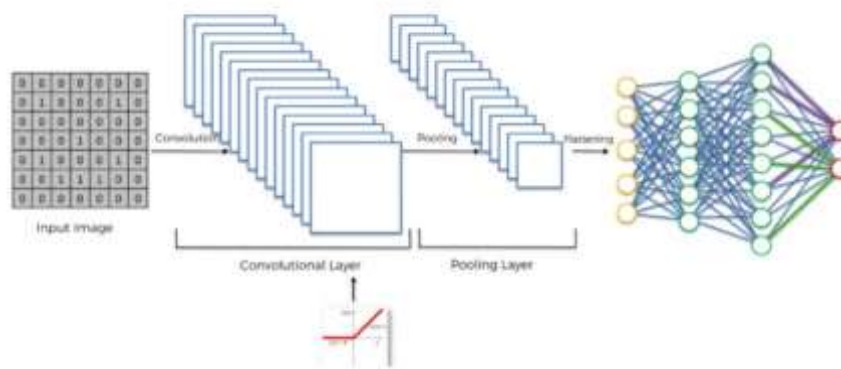


Figura 17. Procedimiento al que se somete una imagen en una red neuronal convolucional. Elaborada por Gomilla (2019)

2.2.8. Software Utilizado

2.2.8.1. Software

Software es un término que se aplica en el campo de la informática y hace referencia a un programa de cómputo, datos y procedimientos que permiten realizar distintas tareas en un sistema informático u ordenador. (Buzon, 2020)

El software viene a ser la parte intangible del ordenador, se considera que es el programa que hace que las demás piezas o hardware del ordenador se comuniquen entre sí o realicen una actividad en concreto.

2.2.8.2. Software Libre

Free Software Foundation (2022) relata que el software libre es el tipo de software que respeta la libertad de una comunidad informática y sus usuarios finales. Significa que los usuarios tienen la libertad de poder ejecutar, copiar, modificar e incluso distribuir mejoras del software.

En base al concepto anterior se puede deducir que el software libre es aquel que se encuentra a disposición de toda la comunidad que le interese, al estar libre se hace referencia a que es gratuito y de código abierto (Que cualquier persona o comunidad puede modificar y crear una versión distinta de la original). Hay que recalcar que el código abierto alude únicamente a la modificación del software existente, ya que, si la persona, comunidad o empresa realiza una modificación de un software libre, ellos pueden decidir si distribuirlo monetariamente o no. Los casos más comunes suelen ser cuando empresas dan una distribución de un software gratis y la remuneración la obtienen a través de actualizaciones, mejoras constantes o incluso mantenimiento. Esto suele ser rentable ya que en empresas muy grandes es más económico implementar software libre y de código abierto en sus equipos que licencias empresariales sumamente costosas.



Figura 1118. Logo del sistema operativo libre de tipo Linux. Adaptado de <https://www.gnu.org/philosophy/free-sw.es.html>. Elaborada por Free Software Foundation

2.2.8.3. Lenguajes de programación

Los lenguajes de programación se pueden dividir en dos tipos muy populares actualmente, esos son los de alto nivel (los más conocidos) y los de bajo nivel (algunas personas consideran que un nivel más bajo es el ensamblador y otros lo categorizan en esta sección).

El lenguaje de programación de alto nivel es más común ya que son mucho más sencillos de comprender, la única desventaja es que es un poco más lento que el de bajo nivel. En cuanto a velocidad se refiere a que un lenguaje de programación es la herramienta que me va a permitir desarrollar software que utilice partes del ordenador o haga una función en específico.

2.2.8.3.1 Python

Python es el lenguaje de programación más popular de la última década, pues desde su creación en los 90's ha venido por una serie de versiones que algunas ofrecían novedades y otras plagadas de errores.

Por lo que se populariza desde el 2008 con la llegada de Python 3.0 donde en inicios era incompatible con versiones predecesoras, pero finalmente se logra una compatibilidad desde

la versión 2.6 para lograr una transición mucho más sencilla. (Challenger, Díaz y Becerra, 2014)

Su popularidad se basa en la versatilidad que ofrece este lenguaje, comenzando desde su sintaxis que es mucho más reducida que el anterior lenguaje más popular C++.

```
#include <iostream>

using namespace std;

int main()
{
    cout << "Hello World" <<
endl;
    return 0;
}
```

Figura 129. Hola mundo en lenguaje C++. Adaptada de <https://www.redalyc.org/pdf/1815/181531232001.pdf> Elaborada por Challenger y cols (2014)

En lenguaje Python todas estas líneas se remplazan simplemente por un print “Hello World” y posterior a ello su compilación. Como se puede notar es un cambio muy notorio y mayor depuración.

También es reconocido mundialmente porque en la actualidad se han creado múltiples interpretes que permiten tener dentro de sí mismo ambientes virtuales distintos, de esta manera poder trabajar con los ambientes específicos para programación tradicional o incluso se adentra en el campo de la Visión artificial. Como lo es Anaconda que cuenta con interpretes dedicados a la inteligencia/visión artificial como lo es Spyder o Pycharm.

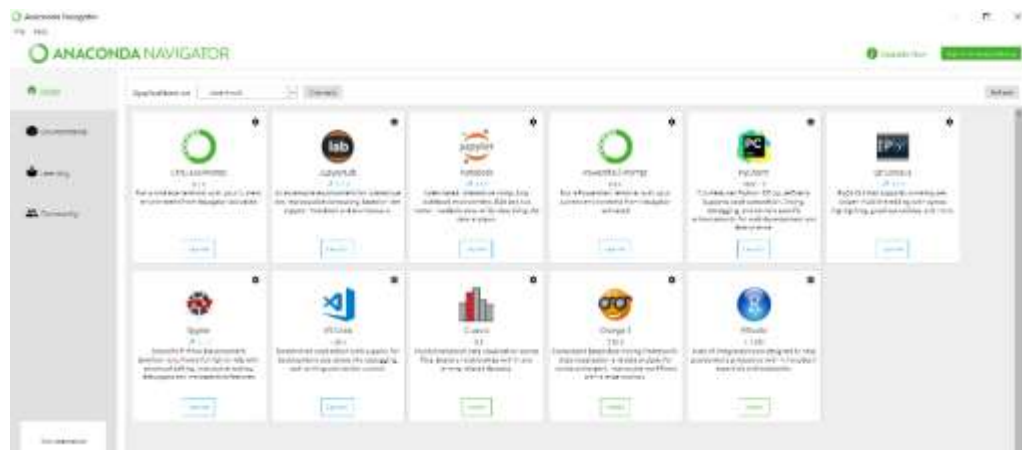


Figura 20. Anaconda Navigator interfaz y posibles interpretes a tener en cuenta. Elaborada por el autor.

Su popularidad yace en que resulta altamente efectivo en el campo de la Big Data y a los Data Scientist. Esto gracias a que se puede manipular, procesar y visualizar grafica de datos que permiten:

- Creación de visualizaciones sobre los datos de alta calidad.
- Creación de dashboards para el posterior análisis de datos.
- Creación de informes automáticos.
- Dispone de herramientas de análisis estadísticos para poder tratar y presentar los datos de manera más óptima.

2.2.8.3.3. Java

Es un lenguaje de programación desarrollado por Sun Microsystems a mediados de los 90's. Gracias a este lenguaje de programación es posible la creación de muchos servicios y aplicaciones pues resulta ser muy eficaz en el desarrollo de aplicaciones móviles mediante IDE como Android Studio, entre otros.

Java es un lenguaje orientado a objetos lo que permite que este lenguaje sea utilizado en el desarrollo de programas o aplicaciones en distintas áreas como lo son la seguridad, base de datos, arquitecturas cliente servidor, webs interactivas, entre otras. (Universidad Nacional Autónoma de México, 2021)

En resumen, Java es muy popular por su versatilidad y por ser multiplataforma lo cual hace que sea más conocido y accesible para los usuarios.

2.2.8.3.4. C++

Es un lenguaje de programación el cual está orientado a objetos. Es una evolución del lenguaje informático C. Este lenguaje es muy potente debido a que sigue usándose en la actualidad a pesar de que su creación data de los 80's debido a que el lenguaje mantiene constantes actualizaciones.

Es un lenguaje utilizado para programación estructurada de alto nivel, idóneo por su sencilla sintaxis para programadores principiantes y sobre todo ya que guarda similitud en algunas líneas o estructura de su código con otros de bajo nivel. Se puede utilizar para aprender los fundamentos de la mayoría de lenguajes de programación.

Workana (2022) indica que C++ se sigue utilizando para el desarrollo de:

- Videojuegos (En Unity se suele usar sintaxis de C++)
- Navegadores de Internet
- Sistemas Operativos
- Base de Datos.
- Creación de bibliotecas o librerías de programación
- App móviles.

Cosas a destacar de este lenguaje de alto nivel es que permite reutilizar bloques de código completos para el desarrollo de nuevos programas. También que al ser multiplataforma es ideal para el desarrollo de aplicaciones. (Workana, 2022)

2.2.8.4. Librerías para Visión Artificial y Aprendizaje automatizado

2.2.8.4.1. OpenCv

OpenCV cuyas siglas significan (Open Source Computer Vision) es una de las librerías gratuitas a implementar en Python y que está relacionada directamente con la visión artificial pues permite la manipulación de imágenes y videos con una gran variedad de opciones para modificarlos. Puede obtener frames o modificar imágenes hasta realizar inferencias en tiempo real mediante una webcam. (Howse, 2020)

2.2.8.4.2. Mediapipe

Mediapipe es un marco que permite construir canalizaciones de aprendizaje automático para procesar datos de series temporales como video, audio, entre otros. Es multiplataforma porque puede ejecutarse desde un ordenador hasta dispositivos limitados como Raspberry Pi de las versiones iniciales. (Kukil, 2022)

Fue desarrollada por Google y compartida para su uso hacia una comunidad interesada en la visión artificial pues tiene la capacidad de poder realizar varios tipos de proyectos de manera rápida y optima sin necesidad de muchos recursos al momento de tener puntos de interés reflejados en el área a investigar. Es decir, se genera un tipo de malla o exoesqueleto que servirá de referencia para el ordenador y la visión artificial que se pretende crear.

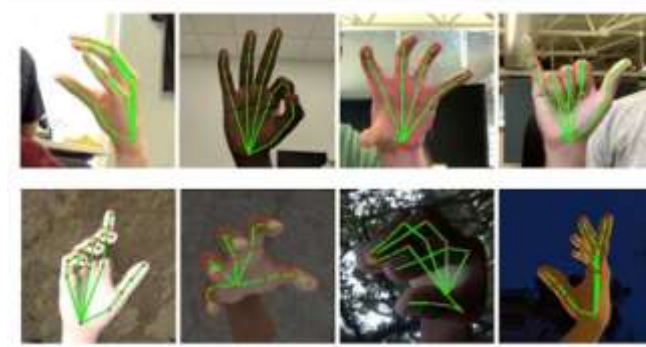


Figura 22. Imágenes de manos sintéticas renderizadas con los 21 puntos de interés obtenida de <https://google.github.io/mediapipe/solutions/hands>. Elaborada por Google (2022)

En la siguiente tabla propuesta por Mediapipe (2022) se describe brevemente los proyectos más reconocidos hasta el momento que ofrece realizar:

Tabla 2. "Proyectos más populares que ofrece Mediapipe"

Proyectos de Mediapipe Más Populares	
Segmentación de selfies	Proporciona máscaras de segmentación para humanos prominentes en la escena.
Malla facial	468 puntos de referencia faciales en 3D con compatibilidad de varios rostros
Seguimiento de manos	21 puntos de referencia en 3D con compatibilidad con varias manos, se basa principalmente en la detección en tiempo real de la palma de la mano con un modelo de puntos de referencia.
Detección y seguimiento de poses humanas	Seguimiento de posturas del cuerpo humano de alta fidelidad, infiriendo hasta 33 puntos de referencia evidentemente en 3D de todo el cuerpo completo. Se basa en los fotogramas que proporcione el video y deberán ser de tipo RGB.
Segmentación del Cabello	Re coloración híper realista del cabello únicamente en tiempo real
Detección y seguimiento de objetos	Detección y seguimiento de objetos en video en un solo pipeline
Detección de rostro	Detector de rostros de bajos recursos que utiliza 6 puntos de referencia y tiene compatibilidad con varios rostros

Seguimiento holístico	Seguimiento simultaneo y en tiempo real de 33 poses, 21 por mano y 468 puntos de referencia para la malla facial
Detección de objetos en 3D	Detección en 3D de objetos cotidianos como zapatos

Información obtenidos de la página oficial de Mediapipe (Fuente: Autor basado en <https://mediapipe.dev/>)

2.2.8.4.3. Tensorflow

Tensorflow es una librería gratuita que puede ser implementada también en interpretes que utilicen Python, es utilizada comúnmente para entrenar y desarrollar modelos de aprendizaje automático (ML). (Estévez, 2022)

2.2.8.4.4. Keras

Según Estévez (2022) Keras es una librería gratuita que puede ser implementada también en interpretes que utilicen Python, es utilizada porque se especializa en el desarrollo de modelos de aprendizaje profundo (Deep Learning) como los son las neuroredes. La diferencia es que esta es una librería que se ejecuta sobre otra librería que en este caso viene a ser Tensorflow.

2.2.8.5. Jetpack Jetson Nano

Es la solución más completa para la creación de aplicaciones en campo de la IA. JetPack SDK proporciona un entorno de desarrollo completo en el que las aplicaciones que tengan que ver con la IA sean optimas y se complemente con el hardware. (NVIDIA, 2022)

Podría resumirse que Jetpack SDK es el sistema operativo que irá en el hardware Jetson Nano Developer Kit y que, al tener una interfaz amigable para el usuario final, será mucho más sencillo la creación de proyectos que tengan como fin la Inteligencia artificial. Actualmente existe la versión JetPack 5.0.1 que se encuentra disponible en la página oficial de Nvidia Developer.

2.2.8.6. Raspberry Pi OS

Es el sistema operativo que puede ser flasheado al micro ordenador Raspberry Pi y consta de una distribución del software libre Linux, su versión Debian. Por lo que es altamente popular es comunidades informáticas ya que la mayoría suele usar esta distribución Debian por su simpleza y efectividad.

2.2.9. Hardware

Es la parte tangible del ordenador o sistema de cómputo, son las piezas que interconectadas entre sí y con ayuda del software lograrán comunicarse y realizar acciones.

2.2.9.1 Jetson Nano Developer Kit

La Jetson Nano como se puede observar en la Figura 23 es un microordenador compacto y sumamente potente que tiene la capacidad de ejecutar múltiples neuroredes en paralelo con aplicaciones del tipo clasificador de imágenes, detección de objetos, segmentación y procesamiento de voz. La gran ventaja sobre la competencia es que contiene una GPU dedicada que facilita este tipo de proyectos y su consumo de energía es de tan solo 5 vatios. (NVIDIA, 2022)



Figura 23. NVIDIA Jetson Nano Developer Kit. Elaborada por NVIDIA (2022).

Las especificaciones del hardware las describe NVIDIA (2022) en la siguiente tabla:

Tabla 3. Especificaciones técnicas de Nvidia Jetson Nano Developer Kit.

Especificaciones Técnicas de NVIDIA Jetson Nano Developer Kit	
GPU	Maxwell de 128 núcleos
UPC	ARM A57 de cuatro núcleos a 1,43GHz
Memoria	4GB LPDDR4 de 64 bits 25,6 GB/s
Almacenamiento	microSD (Se adquiere por separado)
Codificación de video	4K a 30 4x 1080 a 30 9x 720p a 30 [H.264/H.265]

Decodificación de vídeo	4K a 60 2x 4k a 30 8x 1080p a 30 18x 720p a 30 [H.264/H.265]
Cámara	2 carriles MIPI CSI-2 DPHY
Conectividad	Gigabit Ethernet, M.2 Clave E
Monitor	HDMI y Puerto de pantalla
USB	4x USB 3.0, USB 2.0 Micro-B
Otros	GPIO, 2C , 2S , SPI, UART
Mecánico	69 mm x 45 mm, conector de borde de 260 pines

Datos obtenidos de página oficial de NVIDIA (2022) (Fuente: Basado en <https://developer.nvidia.com/embedded/jetson-nano-developer-kit>)

2.2.9.2. Raspberry Pi

Placa reducida o un ordenador reducido de bajo costo que permite crear proyectos informáticos o domotización. En sus versiones más recientes pueden ejecutar proyectos básicos de inteligencia y visión artificial.



Figura 24. "Raspberry Pi 4" adaptada de Amazon. Fuente Amazon.com (2022).

2.2.9.3. Cámaras web

2.2.9.3.1 Cámara web USB

La cámara tradicional de una resolución HD por conexión USB, brinda una resolución decente de 720p y algunas incluyen micrófono incorporado por un módico precio. La cámara web en cuestión es la que se muestra en la Figura 25:



Figura 25. Cámara Web con Micrófono Incorporado de resolución 720p. Adaptada de <https://computron.com.ec/credito/camara-web-rippa-720p-microfono-integrado-usb-cm12>. Elaborado por Computrón.

2.2.9.3.2 Cámara Raspberry Pi

Son un tipo de cámara especial para el micro ordenador “Raspberry Pi” debido a su conexión a través de serial CSI, sin embargo, mucho micro ordenadores cuentan con ese tipo de conexión para este módulo de cámara. Por lo que es compatible con otros tipos de hardware.

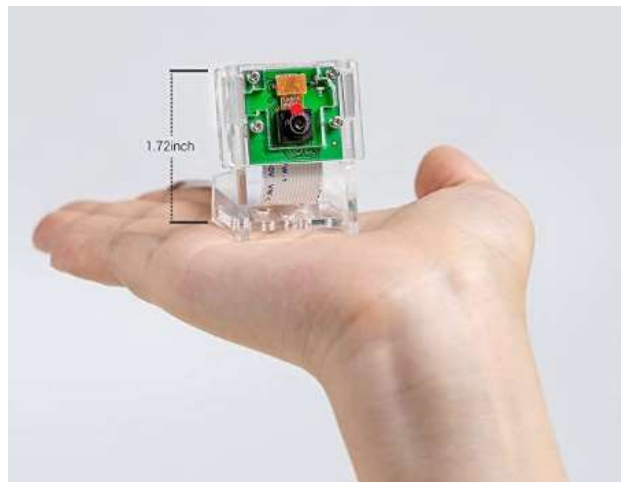


Figura 26. "Módulo de cámara de Raspberry" adaptado de Amazon (2022).

2.2.10. Dactilología de Lengua de señas americano

Lucumi y Marín (2021) mencionan que:

“La dactilología no es más que el nombre que se le da a un alfabeto manual, es decir, un sistema de comunicación que utiliza diferentes posiciones de la mano en algunas ocasiones suele ser movimientos completos y otras la mano estática para representar una letra del abecedario”.

Evidentemente esto existe para poder facilitar el proceso de escritura y comunicación en la comunidad de personas con discapacidad auditiva.

Como se ha mencionado previamente a inicios del trabajo, existen varios abecedarios con distintas señas manuales o kinema. Aunque al menos en el continente americano suelen ser muy similares varios de estos abecedarios, algunos siendo una modificación o inclusión de letras extras, pero teniendo como origen el ASL (American Sign Language). La figura 27 se puede observar la dactilología del ASL.



Figura 27. Abecedario de ASL 2020. Obtenido de <https://quecuriosidades.com/como-funciona-lenguaje-senas/>

2.3Marco Conceptual

2.3.1 Dactilología

Para Herrera (2005) el alfabeto dactilológico es un sistema de comunicación basado en el alfabeto latino, en el cual, cada letra del alfabeto se ve representada manualmente por un movimiento de la mano único y discreto.

Herrera (2005) asegura que la dactilología hace posible reproducir las palabras del lenguaje oral a través de la representación manual de cada una de las letras que la constituyen.

También concluir que el alfabeto dactilológico es una invención para que las personas con discapacidad auditiva se les facilite el aprendizaje del lenguaje oral. Como es costumbre el deletreo se mezcla junto con la lengua de signos de cada país y principalmente se utiliza dicho deletreo para expresar Nombres o incluso palabras que no cuentan con su representación en señas.

2.3.2 Flattening

Estévez (2022) indica que el proceso de flattening se realiza una vez los resultados hayan pasado por las capas más fuertes como lo son la de convolución y Máxpooling. Consiste en acomodar el mapa de características de interés en un vector unidimensional, para finalmente este vector pase a ser los datos de la capa de entrada de la red neuronal artificial lo cual será mucho más optimizado debido a la manera en que se tratan los resultados y más fáciles de procesar para el ordenador.

Por lo que deducimos que esta es una capa que permite una optimización después de tener varios mapas de características resultantes de las capas más complejas como lo son la de Pooling y convolucional, con el fin de que esa cantidad de datos sea tratada eficazmente y en el menor tiempo posible que se permita el ordenador.

2.3.3 Redes Neuronales Convolucionales

Massiris y cols (2018) relatan que:

“El propósito de las redes neuronales convolucionales es extraer todas las características de una imagen y luego usar dichas características para detectar o clasificar los objetos en una imagen. Los parámetros de los filtros que se pueden aprender en estas capas; se ajustarán y optimizarán junto con los componentes de clasificación para minimizar el error de clasificación total”.

Cruz y Gutiérrez (2021) aseguran que las redes neuronales convolucionales utilizan perceptrones multicapa que requieren un preprocesamiento mínimo para entrenar la arquitectura para que realice la tarea de reconocimiento y clasificación de una manera eficaz.

Se puede concluir en base a los conceptos revisados que una red neuronal convolucional es una alternativa altamente eficiente en comparación a algoritmos existentes para reconocimiento de imágenes y videos en el campo de la visión artificial. Entre más profunda sea la red, más neuronas y mayor número de capas suele ser más efectiva.

2.3.4 Pooling

Méndez (2019) indica que la capa de pooling perteneciente a una red neuronal convolucional es aquella capa donde se reducen el número de datos / tamaño de las matrices. Es la capa siguiente a la inicial o convolucional, resultan ser de gran utilidad ya que reducen considerablemente las dimensiones de la matriz bidimensional de la imagen que analiza sin afectar al número de capas de la misma.

El pooling también se conoce como reducción de muestreo debido a que se pierde información al reducir su tamaño.

Existen varias opciones de como implementar el pooling pero las más utilizadas son el Average Pooling y Max Pooling. Donde la primera al reducir encuentra el valor medio del subconjunto de matriz y la segunda encuentra el valor máximo de cada subconjunto de matriz.

2.3.5 Python

“El lenguaje Python es un lenguaje de alto nivel, puesto que puede gestionar sus recursos, y en particular la memoria, mediante un recolector de basura que implementa un contador de referencias”. (Chazallet, 2016, p.53)

Python se podría considerar el lenguaje de alto nivel más popular en la actualidad y desplazando a C++ por ser mucho más eficiente y sencillo de comprender por sus sintaxis para cualquier desarrollador independientemente de su experiencia.

2.3.6 Visión Artificial

“La visión artificial es una disciplina científica formada por un conjunto de técnicas que permiten la captura, el procesamiento y el análisis de imágenes”. (Domínguez, 2021, p.1)

Retomando a Domínguez (2021) menciona que la visión artificial es interpretar las imágenes con el fin de extraer información de utilidad, por ejemplo, saber si delante de una cámara hay alguna persona u objeto de interés. Con la información recolectada se procede a realizar actividades específicas que el desarrollador haya impuesto como puede ser el acceso por reconocimiento facial de personas registradas a una instalación.

La visión artificial puede ser aplicada en una gran variedad de campos debido a las posibilidades que ofrece. Los campos que más explotan este subcampo de IA son: medicina, agricultura, videojuegos, conducción autónoma, clasificación de imágenes, comercio, seguridad, educación, industrial y la lista seguirá en constante crecimiento pues más campos buscan sacar provecho de este avance tecnológico.

2.4 Marco Legal

En el marco legal se describirán artículos de la constitución de la república del ecuador (Encargada de reconocer el derecho que tiene la población ecuatoriana de vivir en un ambiente totalmente sano y equilibrado, también que toda la población tenga acceso a la información pública, entre otros.) y también leyes orgánicas del ecuador (Encargadas de

indicar como cumplir con los derechos estipulados en la constitución) con el fin de presentar la relevancia que tendría el diseño del prototipo de este trabajo de titulación y la importancia que tendrá para las personas con una discapacidad auditiva.

2.4.1. Constitución de la República del Ecuador

La constitución indica que todas las personas son iguales y gozarán de los mismos derechos y oportunidades. Nadie podrá ser discriminado por razones de etnia, lugar de nacimiento, edad, sexo, identidad de género, identidad cultural, estado civil, idioma, religión, ideología, filiación política, pasado judicial, condición socio económica, condición migratoria, orientación sexual, estado de salud, portar VIH, discapacidad, diferencia física, ni por cualquier otra distinción, personal o colectiva, temporal o permanente, que tenga por objeto o resultado menoscabar o anular el reconocimiento, goce o ejercicio de los derechos. (Asamblea Nacional Constituyente del Ecuador, 2008, pág. 11)

Según lo establecido por la constitución se puede deducir que el estado siempre deberá asegurar que absolutamente nadie sea discriminado por lo mencionado antes que puede ir desde algo como la etnia hasta un padecimiento clínico. Este padecimiento incluye también a las personas con discapacidad auditiva que es el grupo de interés en este trabajo de titulación.

2.4.2. Ley Orgánica de discapacidades

Artículo 3.- Fines. - La presente Ley tiene los siguientes fines:

3. Procurar el cumplimiento de mecanismos de exigibilidad, protección y restitución, que puedan permitir eliminar, entre otras, las barreras físicas, actitudinales, sociales y comunicacionales, a que se enfrentan las personas con discapacidad;

4. Eliminar toda forma de abandono, discriminación, odio, explotación, violencia y abuso de autoridad por razones de discapacidad y sancionar a quien incurriere en estas acciones;

5. Promover la corresponsabilidad y participación de la familia, la sociedad y las instituciones públicas, semipúblicas y privadas para lograr la inclusión social de las personas con discapacidad y el pleno ejercicio de sus derechos; (Ley Orgánica de discapacidades, 2012, pág. 6).

Artículo 4.- Principios fundamentales. - La presente normativa se sujeta y fundamenta en los siguientes principios:

8. Accesibilidad: se garantiza el acceso de las personas con discapacidad al entorno físico, al transporte, la información y las comunicaciones, incluidos los sistemas y las tecnologías

de información y las comunicaciones, y a otros servicios e instalaciones abiertos al público o de uso público, tanto en zonas urbanas como rurales; así como, la eliminación de obstáculos que dificulten el goce y ejercicio de los derechos de las personas con discapacidad, y se facilitará las condiciones necesarias para procurar el mayor grado de autonomía en sus vidas cotidianas; (Ley Orgánica de discapacidades, 2012, pág. 6).

Artículo 17.- Medidas de acción afirmativa. - El Estado, a través de los organismos competentes, adoptará las medidas de acción afirmativa en el diseño y la ejecución de políticas públicas que fueren necesarias para garantizar el ejercicio pleno de los derechos de las personas con discapacidad que se encontraren en situación de desigualdad” (Ley Orgánica de discapacidades, 2012, pág. 9).

Todos estos artículos aluden a una cosa en concreto y es como el estado va a velar e implementará estrategias o incentivos que permitan a las personas discapacitadas estar involucrados en todo tipo de situaciones cotidianas, de interés público o privado. Sin embargo, estos artículos están pensados para personas con discapacidades en general, algunas pueden ser un porcentaje pequeño cognitivo, otras personas algo físico como alguna extremidad. El problema se puede presentar en personas con discapacidad auditiva pues la forma de comunicación está limitada y no existe divulgación de la misma, viéndose un poco reducidas sus posibilidades de poder equipararse competitivamente al resto o sentir una inclusión.

2.4.3 TIC's y su uso para personas con discapacidad

El consejo nacional de igualdad y discapacidades emitió una documentación en el 2014 que establece políticas para que personas con discapacidades puedan tener accesibilidad a ciertas actividades o recreaciones sin sentirse excluidos o limitados en ningún sentido, sino lo contrario y poder fomentar más su participación o incluso mejorar la atención que se les puede brindar. Esto en combinación con el uso de las tecnologías de la información hacen que sea posible crear soluciones mediante dispositivos tecnológicos y crear un ambiente más igualitario en espacios recreativos, culturales, entre otros.

Esto puede verse en el siguiente artículo:

“Masificar las actividades físicas y recreativas en la población, considerando sus condiciones físicas, del ciclo de vida, culturales, étnicos y de género, así como sus necesidades y habilidades, para que ejerciten el cuerpo y la mente en el uso del tiempo libre.” (Normas Jurídicas en discapacidad Ecuador, 2014, pág. 83-84)

Capítulo III

Metodología

3.1 Metodología del proyecto

Se busca mediante el prototipo una comunicación efectiva entre personas hipoacúsicas que conocen el lenguaje de señas y personas que desconocen en su totalidad el lenguaje de señas, de manera que se pueda traducir el ASL a través de la placa-ordenador y visión artificial.

El desarrollo del dispositivo será por medio de una Jetson nano kit de desarrollador y cuya programación se basará en lenguaje Python debido a que este último es multiplataforma y mundialmente conocido cuando de aplicaciones de visión artificial se refiere. Al final los resultados de las inferencias realizadas por la cámara utilizada y la traducción se presentarán en pantalla.

Por lo que lo conveniente en este tipo de trabajo es implementar una metodología mixta debido a que se pretende recolectar, analizar y combinar datos de tipo cuantitativos, así como cualitativos en un único estudio o en un programa de investigación de tipo multifase. En este caso un método que sirve para una óptima documentación sería el propuesto por Suarez en 2015 en su tesis doctoral para metodologías de diseño de redes neuronales sobre dispositivos para el procesamiento de señales en tiempo real. La metodología de esta tesis doctoral para creación de redes neuronales indica que aquellas con diseño en aritmética en punto fijo son apropiadas para ser implementadas en circuitos digitales, además de poder ser programables por el diseñador, debido a esto son idóneas para el desarrollo de prototipos como el que se piensa desarrollar.

3.1.1 Modalidad de la investigación

Este proyecto se basa en la investigación de fuentes bibliográficas en su mayoría otras tesis, artículos científicos, ensayos, entre otros tipos que mantengan relación con los objetivos de este trabajo de titulación. Con el agregado de relatar el diseño de un prototipo que, mediante una cámara, programación orientada a la visión artificial y una pantalla que permita presentar la información en tiempo real se puedan traducir gestos a distintas letras que conforman el abecedario dactilológico de interés. Podría decirse que la observación y descripción se ven involucradas

debido a que es necesario para los procesos que conforman el funcionamiento del dispositivo.

3.1.2 Instrumentos de investigación

3.1.2.1 Bibliográfico

Se utiliza este instrumento de investigación, por el simple hecho de que, al existir trabajos con el mismo interés, pero tratados desde diferentes perspectivas, contienen información que es fiable y de paso ayuda a comprender el funcionamiento de librerías de programación, redes neuronales artificiales, fundamentos de la visión artificial y también el uso y función del hardware utilizado.

3.1.2.2 Experimentación

Se pretende verificar el correcto funcionamiento del prototipo, con el fin de verificar que los resultados arrojados por el mismo al detectar las señas estáticas sean lo más acertados posible. A pesar de que existen numerosos trabajos que tratan temas similares al presente proyecto, es necesario tener una etapa de prueba y experimentación ya que así es sencillo recabar información fiable, conocer las limitantes que puede presentar el prototipo y que escenarios son óptimos y cuáles serían de error para futuras actualizaciones.

3.1.2.3 Encuesta

Por medio de encuestas realizadas a las personas de interés (residentes de gye) se pretende ver el desconocimiento sobre los derechos que tienen las personas con discapacidad auditiva. También concordando en la necesidad de un prototipo que permita en cierta medida una comunicación entre personas con discapacidad auditiva que conozcan la lengua de señas con aquellas que no.

3.2 Análisis de factibilidad

3.2.1. Factibilidad Legal

Anteriormente en el capítulo dos se puede observar una sección dedicada al marco legal que constituye este trabajo, se puede visualizar que existen artículos explícitos en la constitución de la república del ecuador y a su vez en el código orgánico de discapacidad que en pocas palabras indica que las personas con discapacidad tienen derecho a instrumentos que permita un acceso y estilo de vida igualitario a sus derecho, para dejar de lado la discriminación y fomentar la inclusión para personas que no puedan hacer uso del lenguaje oral. Por eso su factibilidad radica en esas últimas líneas pues hace posible que las

personas hipoacúsicas que conozcan de lengua de señas puedan darse a entender con aquellas que no tenga conocimientos sobre esta lengua.

3.2.2. Factibilidad Técnica

En la factibilidad técnica se puede observar la razón principal por la cual se implementó un hardware o software específico en el transcurso del desarrollo del prototipo.

Tabla 4. "Comparaciones de lenguajes de programación aptos para visión artificial".

Lenguaje	Ventajas	Desventajas	Sistema Operativo Soportado	Apto para Visión Artificial
C++	Potente lenguaje para creación de sistemas complejos.	No es muy atractivo visualmente y sus IDE mucho menos por lo que ha perdido popularidad en los últimos años	Multiplataforma	Sí, es óptimo en su código, aunque limitado por sus librerías.
Python	Lenguaje más popular de la década pasada y cuenta con entornos de desarrollo muy didactas.	No es el código más rápido al compararlo en ejecución junto a los otros.	Multiplataforma	Sí, cuenta con librerías especializadas en visión artificial.
Lenguaje R	Lenguaje especializado en inteligencia artificial	No es muy popular debido a su complejidad y entornos de desarrollo poco amigables	Multiplataforma	Sí

Java	Se pueden realizar varias funciones aparte de la IA como el desarrollo de apps que se ejecutan en servidores web, aparte de que es el lenguaje más cotizado en ofertas laborales.	No se especializa tanto en IA por lo que la mayoría de funciones se ven limitadas en este tipo de proyectos, aunque con los años se busca implementar más.	Multiplataforma	Sí

Información proveniente de la investigación de la propia tesis. Elaborado por Nelson Alarcón (2022).

El lenguaje de programación a utilizar es el de Python debido a su popularidad y gran variedad de librerías y funciones dedicadas a la IA y visión artificial. Sobre todo, porque mediapipe es compatible con este lenguaje desde su versión 3.8 en adelante.

Una vez analizado el software es imprescindible continuar con la mejor opción para hardware. En este caso queda descartado el microordenador Arduino debido a que ya ha sido utilizado en el pasado en varias ocasiones y para la visión artificial solo era compatible con versiones de Python o de lenguajes de programación que a la fecha están totalmente obsoletas.

Tabla 5. "Comparaciones de microordenadores".

Microordenador	RAM	Procesador	GPU dedicada	Núcleos y Frecuencia	Sistema Operativo
Nvidia Jetson Nano	4GB	ARM A57	Sí	4 núcleos a 1,43GHZ hasta 1,8GHz	JetPack-SDK basado en Linux

Raspberry Pi 4	4GB	ARM Cortex-172	No	4 núcleos a 1,5GHz	Raspbian basado en Linux
----------------	-----	-------------------	----	-----------------------	--------------------------------

Información proveniente de la investigación de la propia tesis. Elaborado por Nelson Alarcón (2022).

En este caso se decidió utilizar el microordenador Nvidia Jetson Nano developer kit que, aunque su precio es mucho mayor que la raspberry, la misma contiene una ventaja abismal al traer una tarjeta gráfica dedicada lo cual aporta bastante en los proyectos de visión artificial y no se ve limitada. Adicional a ello su frecuencia suele ser mayor cuando se le proporciona mayor alimentación por medio de un cargador de barril.

3.2.3. Factibilidad Económica

Los elementos utilizados en la creación del prototipo (hardware) son de fácil acceso pues están disponibles en el mercado y existen variaciones para ahorrar costos. Sus respectivos costos teniendo en cuenta que algunas son aproximaciones debido a que por la temporada pueden variar su precio son:

Tabla 6. "Precio y Cantidades de Hardware utilizado"

Componentes	Cantidades	Costo Unitario	Costo Total
Jetson Nano Developer Kit	1	\$ 350	\$350
Adaptador de cargador europeo a americano	1	\$1	\$1
Cargador 5V-2 ^a	1	\$5	\$5
Cable HDMI	1	\$5	\$5
Pantalla 7 pulgadas	1	\$70	\$70
Cable de tipo micro usb	1	\$5	\$5
Cable ethernet CAT 6	1	\$5	\$5
Tarjeta Micro SD de 32GB	1	\$8	\$8
Costo Total			\$449

Información y precios provienen de la investigación de la propia tesis. Elaborado por Nelson Alarcón (2022).

Adicional a ello hay que mencionar que se utilizó software que se complementa con el prototipo, es importante ya que en muchas ocasiones existen software con versiones o de uso profesional que para su uso es necesario la adquisición de una licencia mensual o de por vida con un costo aproximado y variado según el lugar de residencia en la mayoría de casos.

Se hizo uso de softwares libre lo cual indica que no es necesario el pago constante de licencias, sino lo contrario y su uso es totalmente gratuito para la comunidad de interés.

Tabla 7. *"Precio y Cantidades de Software utilizado"*

Software	Cantidad	Costo Unitario	Total
IDE de			
programación de	1	\$0,00	\$0,00
Python (Pycharm)			
S.O JetPack SDK	1	\$0,00	\$0,00
		Costo Total	\$0,00

Información y precios provienen de la investigación de la propia tesis. Elaborado por Nelson Alarcón (2022).

3.2.4. Factibilidad Operacional

El desarrollo del prototipo está orientado a las personas con algún tipo de discapacidad auditiva, a las cuales se les brindará un medio que les facilitará ser comprendido, permitiendo su inclusión en el ámbito social, para ser específico en ambientes con gran afluencia y de respuesta rápidas dirigida a la ciudad de Guayaquil pues tiene un gran número de este tipo de ambientes ya que por lo mismo se la considera la ciudad del comercio, aparte de que hay un mayor número de concentración de personas con este tipo de discapacidad, generalmente dominan las hipoacúsicas.

Con el desarrollo del prototipo se pretende que haya mayor inclusión y aceptación de las personas con algún problema auditivo y del lenguaje en este tipo de entornos para que puedan desempeñarse de manera adecuada con aquellas personas que desconocen el lenguaje de señas.

3.3 Población y Muestra

3.3.1 Población

Según INEC en el último censo realizado en la ciudad de Guayaquil, arrojó que esta ciudad cuenta con aproximadamente 2'698.000 de habitantes.

3.3.2. Muestra

Para poder obtener una muestra en la cual trabajar, se va a usar la fórmula estadística que permita reducir considerablemente el número de la población y con dicho valor reducido poder aplicar las herramientas de investigación. Dado que la población objetivo es finita, se debe aplicar la siguiente fórmula con el fin de tener el tamaño de la muestra y una parte representativa de Guayaquil:

$$n = \frac{Z^2 * (p * q)}{e^2 * \frac{Z^2 * (p * q)}{N}}$$

Donde:

n = Tamaño de muestra buscado

N = Tamaño de la población

Z = Nivel de confianza

e = Error de estimación máximo aceptado

p = probabilidad de éxito

q = (1-p) probabilidad de fracaso

Por lo que los datos de interés quedan de la siguiente manera:

Tabla 8. *"Datos para obtener la muestra en base la población"*

Datos para la muestra			
N	=		2'698.000
Z	=		0,95 – 1,96
e	=		5%
p	=		50%
q	=		50%
n	=		Por descubrir

Información proveniente de la investigación de la propia tesis. Elaborado por Nelson Alarcón (2022).

Así se tiene:

$$n = \frac{(1,96)^2 * (0,5 * 0,5)}{(0,05)^2 * \frac{(1,96)^2 * (0,5 * 0,5)}{(2'698.000)}}$$

$$n = 385 \text{ personas}$$

3.4 Interpretación y análisis de datos

En este apartado se va a observar y analizar los resultados brindados por la encuesta realizada que tiene como objetivos. Tener conocimientos sobre cuantas familias cuentan con

una persona con discapacidad en su hogar, la forma en la que logran comunicarse, si han encontrado una solución factible y que esperarían en un prototipo de este estilo.

Por ende, para corroborar este trabajo y gracias a la muestra la encuesta fue dirigida a 385 personas de la ciudad de Guayaquil.

Entre estas 385 personas encuestadas es evidente que se va a contar con la opinión de personas de interés como lo son estudiantes de carreras con fines informáticos (Telemática, sistemas, entre otras), personal de atención al cliente en locales de comidas rápidas y personas con discapacidad auditiva.

1. ¿Cuenta con algún familiar/conocido con discapacidad?

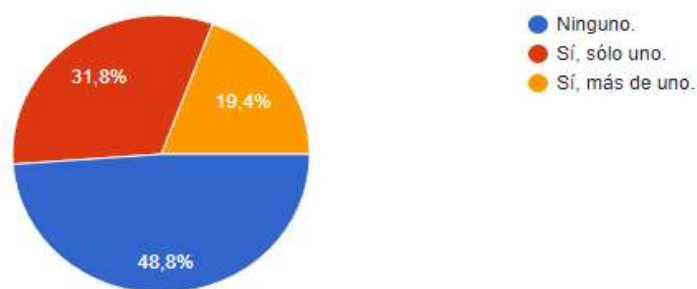


Figura 28. Cantidad de personas con discapacidad en la familia. Información obtenida de la investigación. Elaboración propia.

Análisis: Aproximadamente 187 personas no cuentan con conocidos o familiares con algún tipo o porcentaje de discapacidad.

Por otro lado 123 personas cuentan con algún conocido o familiar con algún tipo de discapacidad.

Finalmente, un porcentaje considerable correspondiente a 75 personas tienen a más de un conocido o familiar con algún tipo de discapacidad.

Conclusión: El CONADIS recalca en sus estadísticas del inicio, los mayores números en cuanto a personas con algún tipo de discapacidad se suelen concentrar en ciudades turísticas y aglomeradas como lo es Gye y esta encuesta lo refleja.

2. ¿Cuenta con algún familiar/conocido con discapacidad auditiva y/o del lenguaje?

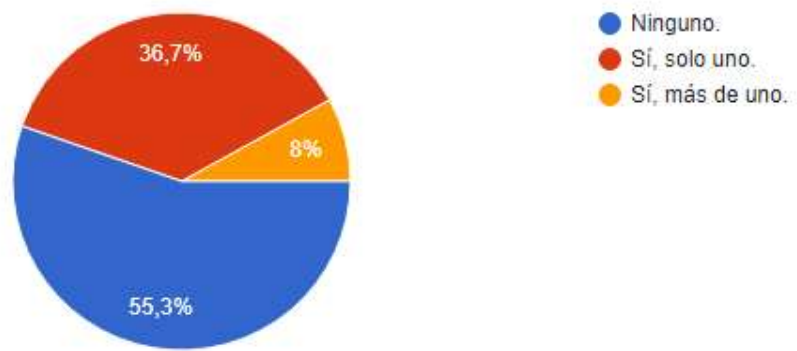


Figura 29. Cantidad de personas con discapacidad auditiva en la familia. Información obtenida de la investigación. Elaboración propia.

Análisis: Más de la mitad de los encuestados (212) no cuentan en sus familias o amistades con alguna persona que tenga algún tipo de discapacidad auditiva y/o del habla.

Aproximadamente 142 personas cuentan con una persona entre su familia o amistades que padece de algún tipo de discapacidad auditiva.

Alrededor de 31 personas cuentan con más de una persona entre su familia o amistades que padece algún tipo de discapacidad auditiva.

Conclusión: Existe un número considerable de personas con discapacidad auditiva entre los familiares o conocidos de los encuestados, por lo que se concluye que en las personas que eligieron “más de una persona entre sus familiares y/o conocidos con discapacidad auditiva” puede deberse a que conocen parejas de sordos mudos como es habitual o como se ha registrado en carreras como en Diseño gráfico de la UG existen numerosos estudiantes con este tipo de discapacidad. Esta pregunta y sus resultados son de gran aporte para el desarrollo de un prototipo que permita la traducción de señas, pues muestra que existe un público objetivo considerable para ello.

3. ¿Con qué frecuencia ha visto usted a una persona con discapacidad auditiva en sitios de gran afluencia?

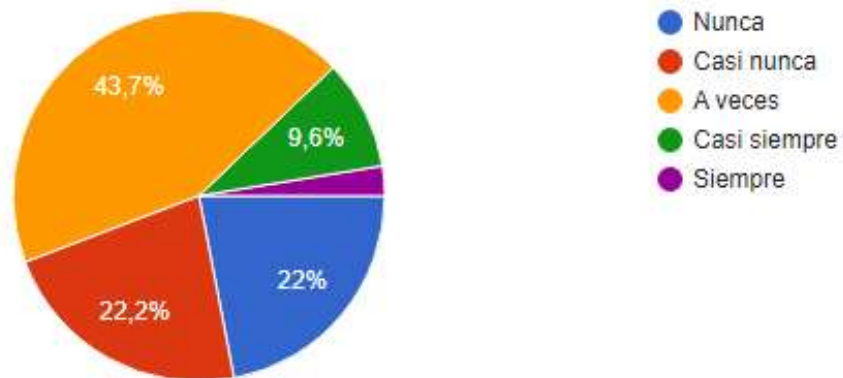


Figura 30. Frecuencia en sitios de gran afluencia. Información obtenida de la investigación. Elaboración propia.

Análisis: 167 personas encuestadas suelen ver con frecuencia (a veces) a personas con discapacidad auditiva en sitios de gran afluencia.

Fueron 86 personas que suelen ver con poca frecuencia (casi nunca) personas con discapacidad auditiva en sitios de gran afluencia.

Las personas que no han visto nunca o admitieron no percatarse de personas con discapacidad auditiva en sitios de gran afluencia fueron 85.

37 personas afirmaron casi siempre estar en contacto con personas con discapacidad auditiva en sitios de gran afluencia

Únicamente 10 personas de las encuestadas afirmaron siempre estar en contacto con este tipo de personas.

Conclusión: Esta pregunta al igual que la anterior sirven como base sólida para el desarrollo de un prototipo que permita una solución efectiva al problema de comunicación. Se puede concluir que aquellas personas que están en contacto con personas con discapacidad auditiva en sitios de gran afluencia casi siempre y siempre es por el hecho de tener familiares cercanos o convivir con ellos diariamente por lo que siempre estarán presentes y por eso la respuesta y su porcentaje.

4. ¿Cómo se comunican con su familiar/conocido con discapacidad auditiva?

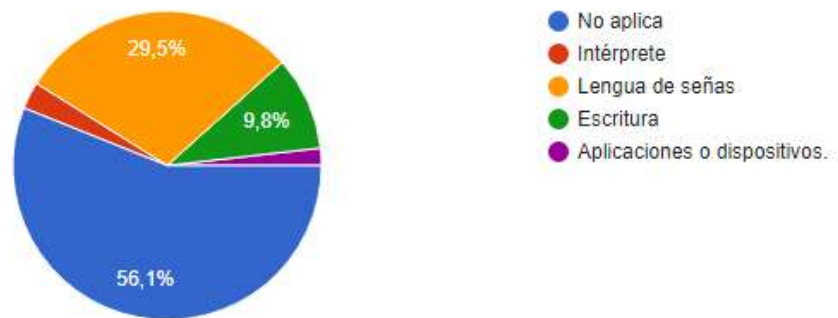


Figura 31. Formas de comunicación. Información obtenida de la investigación. Elaboración propia.

Análisis: Aproximadamente 215 personas indican que no aplica el comunicarse directamente con dicha persona.

Aproximadamente 114 personas de las encuestadas utilizan la lengua de señas como medio de comunicación con personas con discapacidad auditiva.

Solamente 38 personas indicaron comunicarse a través de la escritura.

Las restantes 18 personas se dividen entre intérpretes (7) o aplicaciones (11).

Conclusión: La mayoría de personas utilizan como medio de comunicación la lengua de señas, dentro del país existe la lengua de señas ecuatoriana (LSE) que contiene 30 letras en su abecedario. Concluyendo que la lengua de señas es la comunicación más efectiva con personas discapacitadas dentro del país. Las menos efectivas es por el hecho de que el costo de mantener un intérprete es muy elevado y la mayoría de accesorios o aplicaciones son de origen extranjero y se limitan para ese país.

5. ¿Tiene conocimientos sobre la Inteligencia artificial o visión artificial?

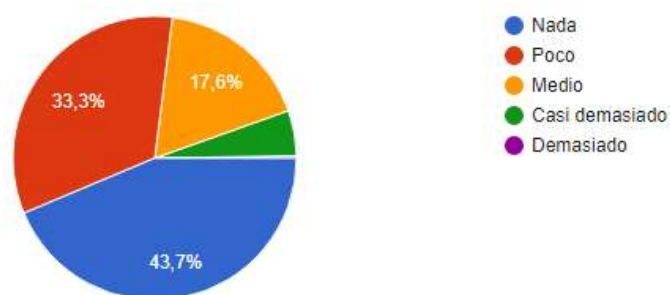


Figura 32. Conocimientos sobre IA/visión artificial. Información obtenida de la investigación. Elaboración propia.

Análisis: Se observó como la mayoría de los encuestados desconocen totalmente sobre este campo de inteligencia artificial. Para ser exactos 168 personas de los encuestados desconocían totalmente sobre el tema.

Aproximadamente 129 personas tienen los conocimientos mínimos sobre el campo, aunque se lo adjudicaban netamente a los videojuegos o robots autómatas.

De todos los encuestados 68 personas tenían un conocimiento medio sobre la IA/Visión artificial y solamente 20 personas conocían casi demasiado sobre el tema.

Conclusión: Dentro del país el tema sigue siendo poco explorado y considerado innovador. Por lo que muchas carreras ni siquiera tratan superficialmente estos temas, dando como resultado los porcentajes de la encuesta que se presenta en pantalla. Se concluye que aquellas personas de conocimiento medio (68) o demasiado (20) eran estudiantes pertenecientes a carreras relacionadas con la informática por lo que conocen de manera un poco más profunda sobre el tema. Sin embargo, no se encontraron personas totalmente especializadas en el tema ya que como se ha mencionado es un tema muy tratado en países de primer mundo por lo que dentro del país suele ser algo muy complejo de comprender y poco atractivo a pesar de las grandes ventajas que trae consigo.

6. ¿Tiene conocimientos sobre cómo la IA/visión artificial puede crear soluciones para la traducción de señas en tiempo real?

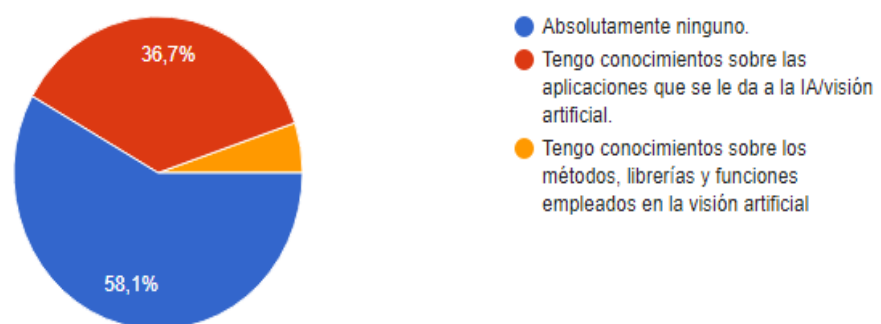


Figura 33. Conocimientos sobre IA y sus soluciones. Información obtenida de la investigación. Elaboración propia.

Análisis: Un total de 224 desconocían totalmente que la Inteligencia Artificial o visión artificial podría estar involucrada directamente en la solución de problemáticas de este estilo.

Aproximadamente 141 personas tienen una idea de las aplicaciones que se le dan a la IA, aunque muchas nuevamente caen en robots o inteligencias de videojuegos.

Únicamente 20 personas respondieron tener conocimientos un poco más específicos sobre soluciones que brinda la visión artificial

Conclusión: Esta pregunta se relaciona con la anterior, pero profundizando un poco más, donde se concluye que los conocimientos medios que se tiene sobre el campo más hacen referencia a la IA que a la visión artificial pues el primero es más extenso y popular. Los que tienen conocimiento especializado en la visión artificial se le atribuye a personas que estudien carrera que incluyen materias a fines como lo son “cibernética e inteligencia artificial” que se tratan métodos, teoría y librerías.

7. ¿Conoce alguna aplicación o dispositivo que permita la traducción de señas en tiempo real?

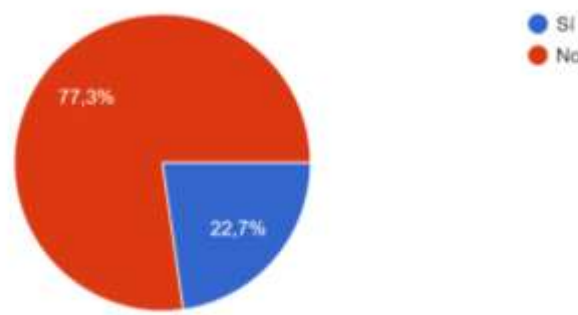


Figura 34. Conocimiento sobre apps de traducción. Información obtenida de la investigación. Elaboración propia.

Análisis: En esta pregunta un total de 298 personas desconocen sobre un prototipo especializado y orientado a la traducción de la lengua de señas dentro del país y aproximadamente 87 tienen noción de algún dispositivo o aplicación que lo permita.

Conclusión: En esta pregunta se buscaba información sobre si habían escuchado ya sea por artículos científicos, noticias, corporaciones, entre otros. Se concluye que casi el 80% no ha escuchado algún proyecto relacionado con la traducción de señas. Mientras que un poco más de 20% de personas sí habían presenciado alguna vez publicaciones o noticias de este estilo.

8. ¿Cómo le resultaría más cómodo un dispositivo traductor de lengua de señas en sitios de gran afluencia?

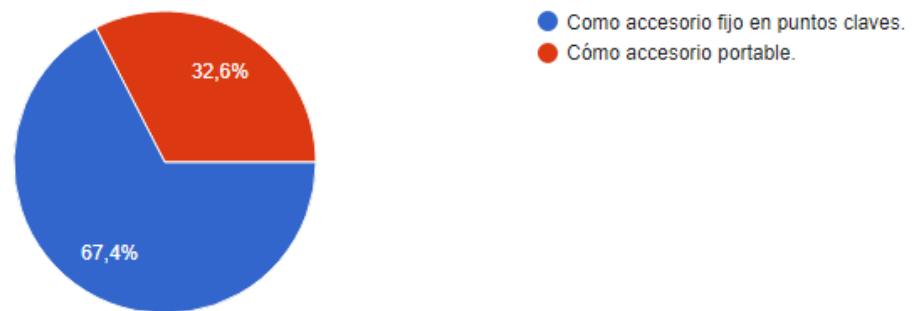


Figura 35. Preferencia del prototipo en sitio de gran afluencia. Información obtenida de la investigación. Elaboración propia.

Análisis: La mayoría de personas (260) prefirió que en sitios de gran afluencia haya un punto fijo en lugares donde se lo requiera puesto que es incómodo el utilizar accesorios como guantes molestos para la traducción.

Un total de 125 personas prefieren un accesorio portable como lo son los guantes traductores.

Conclusión: La mayoría prefiere que un dispositivo con cámara realice el trabajo completo. Lo cual avala la importancia del desarrollo de un prototipo que no deba implementarse como accesorio, sino que dentro de una distancia o rango prudente realice la acción. Se concluye que el restante porcentaje ve más afinidad por un accesorio por lo llamativo e innovador que puede observarse.

9. ¿Qué preferiría en un dispositivo que pueda traducir la lengua de señas a texto en sitios de gran afluencia?

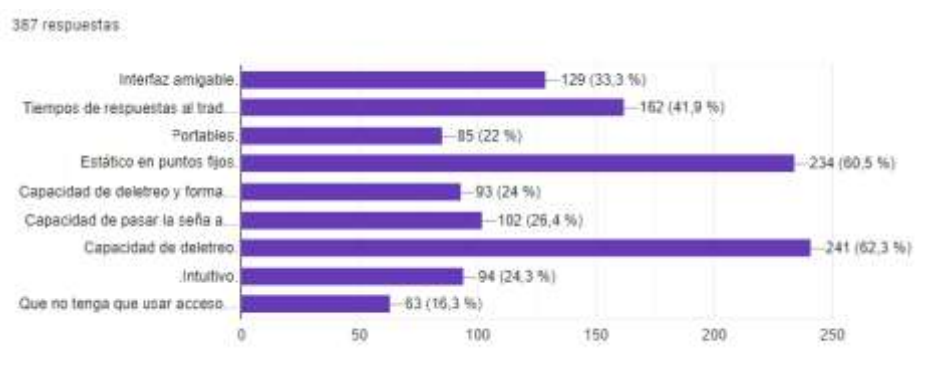


Figura 36. Requisitos para el sistema. Información obtenida de la investigación. Elaboración Propia.

Análisis: En la figura 36 se ven detalladamente lo que los usuarios buscan en un dispositivo de detección en tiempo real de señas y su traducción en donde lo que coincide la mayoría es en que se desarrolle una interfaz amigable, que el prototipo sea estático, permitiendo un deletreo cuyo tiempo de traducción sea lo más rápido posible. Son las 4 características principales que debe cumplir el prototipo.

Adicional a estas características existen varias en las que se coincidieron como al traducir pasar el texto a un audio o incluso la formación de palabras mediante deletreo. Cosas que se buscaran implementar a lo largo del desarrollo una vez tratada las 3 características principales más solicitadas y en caso de no poder dejarlo como recomendación para trabajos futuros.

Conclusión: Las principales características que busca un usuario en un prototipo que permita la traducción de lengua de señas en sitios de gran afluencia donde los tiempos de sean bajos son:

- Capacidad de deletreo.
- Estático en puntos fijos.
- Tiempos de respuestas rápidos.

3.5 Desarrollo de la propuesta

3.5.1 Requerimientos Funcionales

Tabla 9. "Requerimiento funcional #1".

Id	RQF-01
Nombre	Reconocimiento de mano del usuario
Tipo	Requerimiento funcional
Prioridad	Alta
Pre-rrequisito	N/A
Descripción	El sistema deberá reconocer cuando el usuario coloca su mano en el punto de interés y verificar si la misma es izquierda o derecha
Entrada	Librería hand detector de cvzone
Proceso	Detección de mano
Salida	Registro de la mano del usuario

Información proveniente de la investigación de la propia tesis. Elaborado por Nelson Alarcón (2022).

Tabla 10. *"Requerimiento funcional #2".*

Id	RQF-02
Nombre	Seguimiento de la mano del usuario
Tipo	Requerimiento funcional
Prioridad	Alta
Pre-rrequisito	N/A
Descripción	El sistema deberá ser capaz de seguir la mano en tiempo real siempre y cuando se encuentre dentro del rango establecido
Entrada	Función handtracking proveniente de handdetectotr de cvzone
Proceso	Seguimiento de mano
Salida	Posición de la mano del usuario

Información proveniente de la investigación de la propia tesis. Elaborado por Nelson Alarcón (2022).

Tabla 11. *"Requerimiento funcional #3".*

Id	RQF-03
Nombre	Clasificación de imágenes realizadas por el usuario
Tipo	Requerimiento funcional
Prioridad	Alta
Pre-rrequisito	N/A
Descripción	El sistema deberá clasificar cada una de las letras del lenguaje ASL mediante la mano del usuario
Entrada	Mano del usuario y calificación de mediapipe
Proceso	Clasificación de letras del abecedario
Salida	Creación de dataset de prueba y validación

Información proveniente de la investigación de la propia tesis. Elaborado por Nelson Alarcón (2022).

Tabla 12. *"Requerimiento funcional #4".*

Id	RQF-04
Nombre	Predicción de cada signo del alfabeto de señas realizado por el usuario
Tipo	Requerimiento funcional
Prioridad	Alta
Pre-rrequisito	N/A
Descripción	El sistema deberá predecir la seña realizada por el usuario en tiempo real y con un porcentaje de eficiencia superior al 70%

Entrada	Mano del usuario y modelo entrenado por la RNC
Proceso	Predicción de señas
Salida	Seña clasificada y mostrada en pantalla

Información proveniente de la investigación de la propia tesis. Elaborado por Nelson Alarcón (2022).

3.5.2 Requerimientos no Funcionales

Tabla 13. *"Requerimiento no funcional".*

Ref	Nombre	Descripción
RN001	Tiempos de respuesta aceptables	Que no tengan tiempos elevados al momento de predecir en pantalla la seña mostrada
RN0002	Utilización poco compleja	Que el usuario no deba realizar un proceso tedioso cada vez que desee utilizar el prototipo

Información proveniente de la investigación de la propia tesis. Elaborado por Nelson Alarcón (2022).

3.5.3 Entidades del sistema

En la siguiente tabla se observan las entidades a ser implementadas en el sistema. Cada una de las entidades va a estar con su descripción que va a permitir tener una comprensión mayor respecto a la forma de operar de cada una.

En este caso se puede observar las 5 entidades correspondiente a los usuarios, el dispositivo en cuestión y los procesos a realizar.

Si bien es cierto existen entidades en segundo plano, pero las principales son las que se detallan en la tabla 7, una vez establecidas se puede usarlas como base para el desarrollo de los diagramas de casos de usos y tener una definición de su implementación.

Tabla 14. "Entidades principales del sistema"

Entidad	Descripción
Persona	El encargado principal que va a usar el prototipo a desarrollar
Prototipo	Dispositivo encargado de realizar la captura de seña y su predicción en tiempo real
Procesamiento	Función de alta prioridad que permite la segmentación, ajuste de tamaño adecuado

Reconocimiento

de las imágenes para optimización en tiempos de respuesta.

Entidad encargada de la clasificación de seña.

Seña final

Representación escrita de la seña por pantalla

Información proveniente de la investigación de la propia tesis. Elaborado por Nelson Alarcón (2022).

3.5.4 Diagramas de casos de usos**3.5.4.1 Diagrama de caso de uso reconocimiento de manos**

En la figura 37 se observa como el usuario puede interactuar con el prototipo y este a su vez reconocerá cuando una mano del usuario esté en el rango de predicción.

Se tiene que pasar previamente por algunos procesos antes de que el prototipo pueda reconocer eficazmente la señal articulada por el usuario.

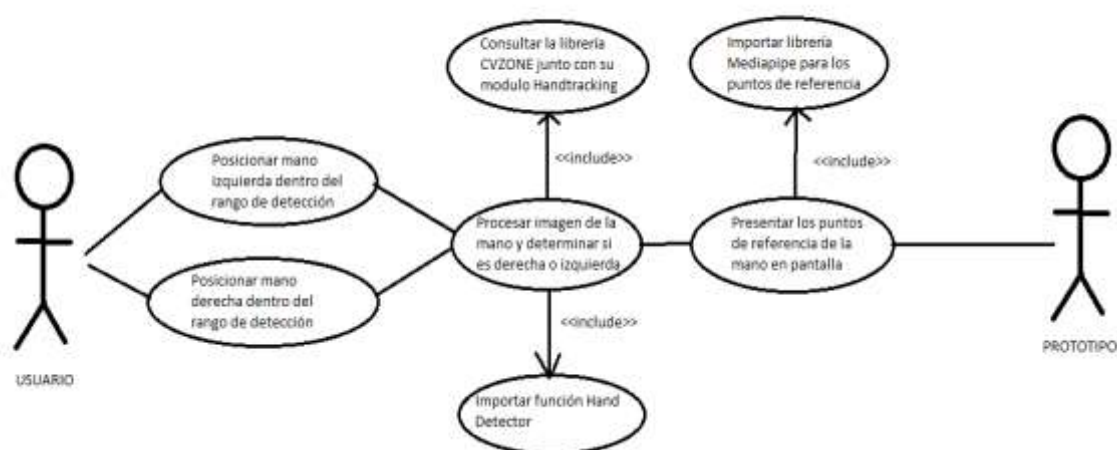


Figura 37. Diagrama de caso de uso de reconocimiento de manos". Información obtenida de la investigación. Elaboración propia.

3.5.4.2 Diagrama de caso de uso seguimiento de manos

En la figura 38 se observa como mediante un diagrama de caso de uso se especifica el proceso de seguimiento de la mano del usuario

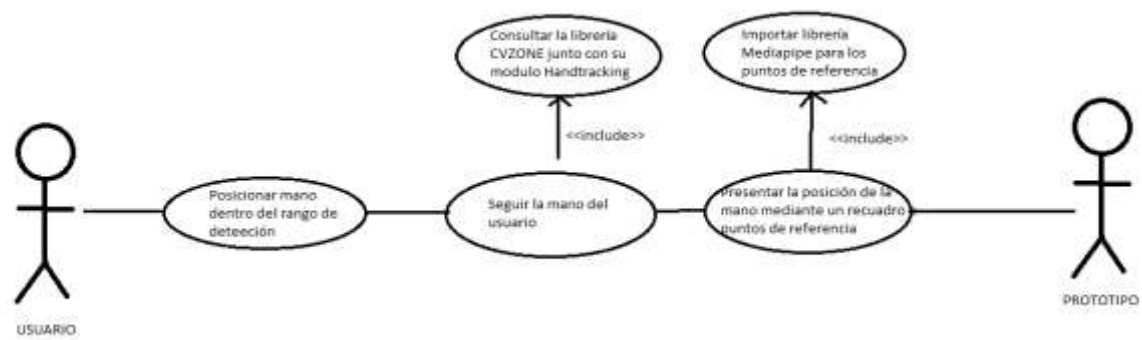


Figura 38. Diagrama de caso de uso seguimiento de manos. Información obtenida de la investigación. Elaboración propia.

3.5.4.3 Diagrama de caso de uso predicción de seña

En la figura 39 se puede observar de forma general, el proceso de predecir la seña realizada por el usuario.



Figura 39. Diagrama de caso de uso predicción de seña. Información obtenida de la investigación. Elaboración propia.

3.6 Desarrollo del prototipo

3.6.1. Creación del dataset del abecedario dactilológico

El dataset es previamente elaborado, el mismo está constituido por 26 letras correspondientes al alfabeto americano o conocido como ASL, estas letras son: A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z. Donde las letras Z y J hay que recalcar que son dinámicas, es decir que en sus señas hay movimientos por lo cual la

alternativa para adjuntarlas al dataset es volverlas estáticas de la manera que el final de la seña estática sea reconocido como la letra final.

Cada letra contiene 500 imágenes para su validación y entrenamiento. Es decir, se va a observar en el dataset para el entrenamiento de la red con un total de 13,000 imágenes. Es importante recalcar que entre más datos o imágenes se proporcionen, mejor será su predicción.

Para la recolección de las imágenes se puede utilizar una cámara web, la cual podrá utilizarse mediante la librería CVzone en Python.

3.6.2. Estandarización de las imágenes

Las imágenes tienen un tamaño de 300x300 píxeles de alto y ancho. Es necesario saber que al momento de recolección de imágenes por defecto el sistema lo realiza en escalas de grises por lo que para presentarlas al final es necesario que los resultados se cambien de escalas grises a RGB mediante funciones ofrecidas por la librería cv2. Por lo tanto, esta vendría a ser la primera etapa del desarrollo donde se programa el algoritmo que realizará los siguientes puntos:

- a) Recorrer las 26 carpetas correspondientes a las 26 letras de interés, en caso de no existir dicha carpeta, se crea automáticamente. Luego de la creación de la carpeta se procede con la toma de imágenes donde se deberá realizar la seña de la letra correspondiente a la carpeta creada.
- b) Las imágenes se almacenarán con un nombre distinto cada una y parará únicamente cuando se oprima una tecla de finalización o cuando el contador que inicialmente está en 0 llegue a 500. Las imágenes por almacenar al estar en escalas de grises, se pasarán a RGB. Una vez realizado la recolección que se recomienda hacer las señas cerca de la cámara, con buena iluminación y un fondo monocromático, para obtener un dataset de este estilo.

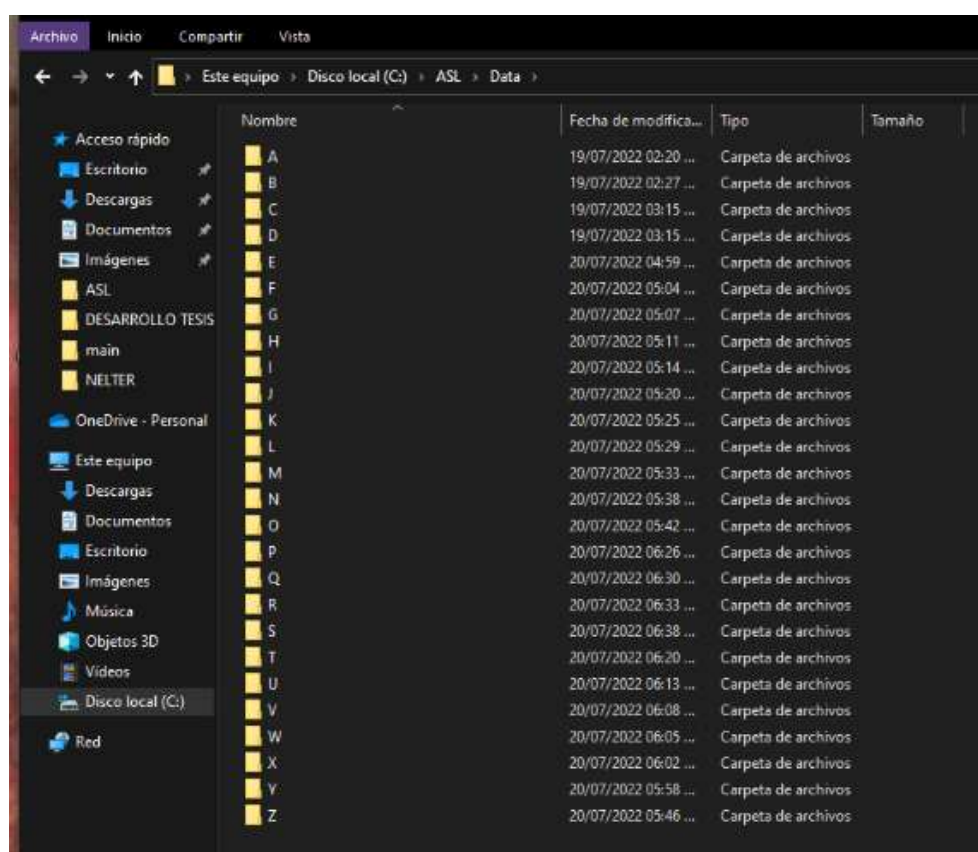


Figura 40. Resultado de algoritmo de recolección de datos. Elaboración propia.



Figura 41. Resultados de recolección de letra A. Elaboración propia.



Figura 42. Creación de Alfabeto basado en ASL. Fuente elaboración propia.

3.6.3. Preparación de los conjuntos de testeo y validación

Es necesario saber que para una red neuronal es necesario destinar imágenes tanto para entrenamiento o testeo y para la posterior validación. Por lo que la mayoría recomienda que del dataset creado, el 80% a 85% de las imágenes total sean dedicadas para entrenamiento y el porcentaje restante para validación. Es decir, de las 13.000 imágenes, 10.400 serán dedicadas al entrenamiento y las restantes 2600 para su validación. El código se puede observar en el ANEXO N° 1.

3.6.4. Desarrollo de la Red Neuronal Convolutiva

Para el desarrollo de la red neuronal utilizada en el prototipo, se tomó de inspiración la arquitectura AlexNet. Debido a que la misma en su etapa de convolución cuenta con 5 capas de convolución, 5 capas de pooling del tipo max-pooling y 5 capas de normalización por lotes, así como la función de activación ReLU. Dicha función de activación está presente al final de cada capa convolutiva y no es más que la formación de modelos de aprendizaje profundo una vez el entrenamiento haya culminado.

Se utiliza inspiración de esta arquitectura ya que fue la primera en hacer notoria las ventajas de las capas de convolución y la que mejores resultados dio en su momento. Aparte de que la considero muy completa para lo que se desea desarrollar ya que se limita a letras estáticas.

3.6.4.1 RNC elaborada

Se desarrolló una RNC de tipo secuencial de entrenamiento con 26 clases correspondientes a cada letra, 5 capas de convolución con su activación al final de la misma con la función ReLU que me permite guardar dichos modelos. También 5 capas de max-pooling para obtener mapas de datos provenientes de las imágenes y tener información específica a utilizar.

Como las imágenes contienen varias dimensiones y por lo mismo se vuelve muy complicado para el sistema obtener características de imágenes con más de una dimensión, hay que crear una capa especialmente para el aplanado de imágenes con el propósito de obtener una única dimensión por cada imagen de interés para de esta manera tener toda la información de interés de una imagen profunda en una plana con escala de grises.

Finalmente, como el modelo cuenta con la información, pero no de una manera optimizada, se procede a someter a la red a través de una capa optimizadora que no es más que pasar toda la información por una función especializada en optimizar llamada Adam, la cual la brinda la librería Tensorflow y a su vez para hacerla mucha más óptima se le puede adjuntar un radio de aprendizaje, el radio en este caso será de 0.0005, es decir, este será el ajuste de la red neuronal que permitirá acercarse a una solución óptima. Por lo que el valor inicial irá desde 0.0005 incrementando su valor por el mismo número y dicho proceso culmina cuando se llegue a 1. La única desventaja es que es un proceso que suele tardar demasiado, uno puede variar el valor de aprendizaje, pero entre más pequeño sea, mayor será su precisión.

Una vez finalizado el proceso anterior, se prosigue con toda la información obtenida otorgarla a la red creada para su entrenamiento, donde cada RNC debe contar con ciertos parámetros a establecer como lo son las iteraciones (epochs), pasos (steps per epoch), validación de imagen y validación de los pasos. Cuando se hayan establecido estos parámetros se puede guardar la información en un archivo de extensión .h5 con la función `cnn.save` con el nombre a elección para utilizarlo en la predicción en tiempo real. También

se hace mención de utilizar la función métrica que me permitirá visualizar el porcentaje de aciertos de la máquina con respecto a la información que le he proporcionado.

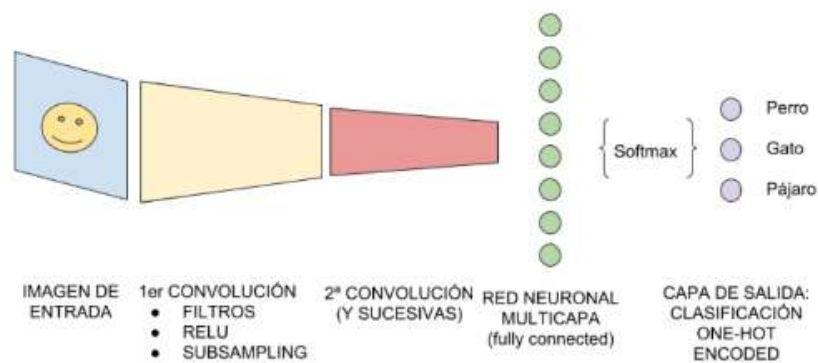


Figura 43. Forma general de una RNC. Adaptada de www.juanbarrios.com

Por lo que para ser más técnicos el proceso de la RNC es:

- a) Se establece la ruta o path de las carpetas que contienen las imágenes de validación y entrenamiento
- b) Se establecen valores y parámetros que se utilizarán a lo largo de la red neuronal convolucional como lo son el radio de entrenamiento, filtros convolucionales, iteraciones, batches o lotes de imágenes, filtro de pooling, entre otros...
- c) Se realiza un preprocesamiento de la imagen en el cual las imágenes se les puede virar, extender, minimizar entre otras modificaciones de imágenes, con el fin de que se cree más información de imágenes internas en base al dataset creado al principio con el objetivo de que la red tenga una gran variedad de manos y pueda predecir mejor.
- d) Establecer y crear la red convolucional de tipo secuencial mediante la línea `cnn = sequential()`
- e) Agregar la primera capa que contiene el primer filtro convolucional de 32 y de pooling (2,2), culminando con la función ReLU.
- f) Luego se repite el proceso con los filtros y capas de pooling en la segunda capa de convolución y sucesivas como se muestra en la figura 34. Con la diferencia que los parámetros de los filtros convolucionales y el tamaño de los mismos irán variando.
- g) Se procede con la obtención de una imagen profunda o con varias dimensiones que pasará por un aplanamiento otorgado por Flatten que permitirá tener toda la información de la imagen tridimensional en una con una sola dimensión haciendo que todo sea más óptimo para el sistema.
- h) Optimizar el modelo con la función Adam y otorgar parámetros que permiten visualizar el porcentaje de precisión en cada iteración.

i) Finalmente, con toda esta información realizar el entrenamiento de la red con parámetros a elección de épocas e iteraciones. Una vez terminado el entrenamiento se puede obtener dos archivos de modelos y pesos con extensión .h5 que serán utilizados en las inferencias en tiempo real.

Por lo que el resultado final un poco más específico y se basa en la arquitectura AlexNet quedaría representado gráficamente de la siguiente manera:



Figura 44. Representación gráfica de arquitectura original AlexNet. Elaborado por Estévez (2022).

Para mayor comprensión de los puntos tratados, se adjunta el ANEXO N° 2 que contiene el código en cuestión.

3.6.5. Entrenamiento mediante una RNC

El proceso de entrenamiento puede tomar mucho o poco tiempo dependiendo de la cantidad de filtros y capas de convoluciones, así como el número de neuronas que contenga la red. Se puede afirmar que el tiempo de entrenamiento y precisión es proporcional al número de filtros, capas y neuronas implementados.

Una vez creada o en este caso programada la red, se puede observar en la *figura 45* y *figura 46* como va aumentando el porcentaje de precisión.

```
Epoch 1/30
300/300 [=====] - 73s 233ms/step - loss: 2.5407 - accuracy: 0.2467 - val_loss: 1.7265 - val_accuracy: 0.1800
Epoch 2/30
300/300 [=====] - 62s 206ms/step - loss: 1.5398 - accuracy: 0.3700 - val_loss: 1.5268 - val_accuracy: 0.4733
Epoch 3/30
300/300 [=====] - 64s 213ms/step - loss: 1.0657 - accuracy: 0.6033 - val_loss: 0.9100 - val_accuracy: 0.7200
```

Figura 45. Inicio de entrenamiento con precisión menor al 20%. Elaboración propia

```

Epoch 26/30
300/300 [=====] - 60s 200ms/step - loss: 0.0985 - accuracy: 0.9667 - val_loss: 0.6927 - val_accuracy: 0.8200
Epoch 27/30
300/300 [=====] - 61s 202ms/step - loss: 0.1189 - accuracy: 0.9567 - val_loss: 0.7261 - val_accuracy: 0.8567
Epoch 28/30
300/300 [=====] - 61s 202ms/step - loss: 0.0688 - accuracy: 0.9767 - val_loss: 1.3506 - val_accuracy: 0.8867

```

Figura 46. Iteraciones finales con precisión por encima del 80%. Elaboración propia.

Cómo se especifica en el inciso anterior, al culminar el entrenamiento de la red, se crean archivos con extensión .h5 que van a servir para únicamente llamar a esos archivos con modelos incluidos y no tener que entrenar una red en cada compilación.

te equipo > Disco local (C:) > LENGUAJE-SENAS

Nombre	Fecha de modificación	Tipo	Tamaño
.idea	05/09/2022 11:50 a. m.	Carpeta de archivos	
Fotos	05/09/2022 11:48 a. m.	Carpeta de archivos	
venv	05/09/2022 11:28 a. m.	Carpeta de archivos	
PC Inferencia_tiempo_real	05/09/2022 11:41 a. m.	JetBrains PyChar...	6 KB
Modelos.h5	21/07/2022 01:07 a. m.	Archivo H5	2,408 KB
Pesos.h5	21/07/2022 01:07 a. m.	Archivo H5	2,408 KB
PC Recolección_de_data	05/09/2022 11:41 a. m.	JetBrains PyChar...	4 KB
PC RedNeuronalConvolutional	05/09/2022 11:41 a. m.	JetBrains PyChar...	5 KB

Figura 47. Creación y visualización de los pesos y modelos creados por la RNC. Elaboración propia.

3.6.6. Instalación del S.O en el ordenador de placa

Una vez se obtiene la Jetson Nano, se procede a instalar su sistema operativo JetPack SDK basado en Linux, existen varios tipos de S.O para este micro ordenador, sin embargo, el jetpack sdk ya contiene en su instalación herramientas que van a ayudar con la visión artificial. Como lo son CUDA o Python instalado directamente.

Los pasos a realizar para la instalación del sistema operativo en la Jetson Nano se puede observar detalladamente en el **Anexo N° 7** y son:

1. Para iniciar se debe formatear la memoria SD con un software especial para Windows llamado “SD card formater” el cual se obtiene de la página oficial de primeros pasos con jetson nano.
2. El siguiente paso es descargar el software BalenaEtcher para poder bootear el sistema operativo de JetPack SDK en la SD que actuará como disco duro de almacenamiento.
3. Se descarga el S.O de la página oficial de Nvidia Jetson Nano, luego con BalenaEtcher, se selecciona “Select Image”, para luego elegir el ISO del S.O.

4. Posteriormente con “Select Drive” en el que se va a seleccionar la tarjeta SD que ya se formateó previamente.
5. Luego hay que presionar en “flash” y esperar que el proceso de booteo culmine de manera exitosa.
6. Posterior a ello se inserta la sd en la ranura específica para la misma que tiene la placa-ordenador. Una vez encendida la Jetson Nano, automáticamente va a comenzar la configuración del sistema.
7. Como es habitual al momento de instalar un S.O, se configura región, hora, nombre de usuario, entre otros.
8. Cuando termina la configuración inicial, se va a reiniciar y aparecerá el logo de NVIDIA para finalmente aparecer el sistema operativo JetPack SDK en nuestra Jetson Nano.

3.6.7. Instalación de versión de Python superiores

La versión de Python que viene por defecto instalado es la 2.7, como todas las versiones previas a la 3 al día de hoy son obsoletas. Por lo que se necesita actualizar mediante la consola a la última versión disponible para este sistema basado en Ubuntu.

En la consola del sistema hay que otorgar los permisos para actualizaciones e instalaciones de las versiones más recientes de Python disponibles.

- \$ sudo apt-get update
- \$ sudo apt-get install python3.6.

El inconveniente con esta versión es que para implementar la librería mediapipe se necesita de la versión de Python 3.8 en adelante y para esta placa ordenador su arquitectura y el sistema basado en Ubuntu solo hay disponible hasta la versión 3.6 de Python. Por lo que para poder acceder a versiones superiores es necesario instalarlo mediante repositorio general de Python o conocido como método PPA de serpientes muertas. Que lo que hace es acceder a una plataforma que contiene las versiones más actualizadas junto con los paquetes adicionales que puedan ser necesarios.

El primer paso es actualizar Ubuntu con los siguientes comandos en la consola:

- sudo apt update && sudo apt upgrade

El Segundo paso es la instalación previa de requisitos para PPA:

- sudo apt install software-properties-common -y

El Tercer paso es agregar el deadsnakes/ppa a la lista de fuentes de paquetes APT mediante el comando:

- `sudo add-apt-repository ppa:deadsnakes/ppa -y`

Una vez importado el repositorio, se ejecuta una actualización APT para actualizar su administrador de paquetes para reflejar el nuevo PPA importado.

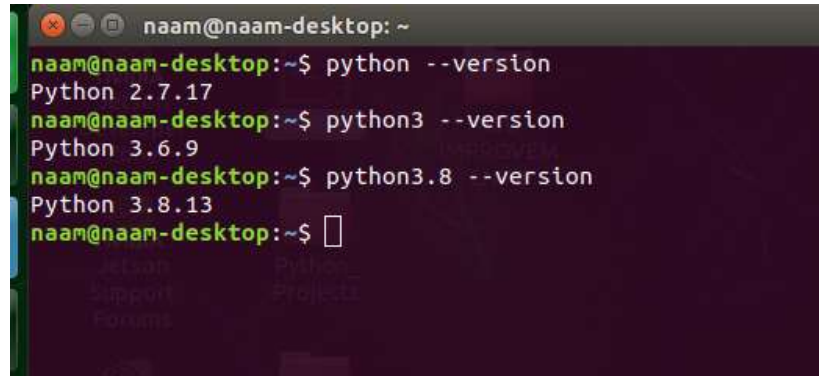
- `Sudo apt update`

Una vez actualizado se puede instalar Python en su versión 3.8 con el comando:

- `sudo apt install python3.8 -y`

Ya con la versión mínima para poder ejecutar funciones e importar módulos de la librería pip, se va a poder ejecutar sin problemas el programa y modelo a obtener gracias a la RNC.

Se puede observar las versiones instaladas con el comando `--version` en la consola. Si se desea la versión por defecto se utiliza Python, si se desea la versión actualizada en inicios se usa python3 y finalmente la versión específica obtenida mediante python3.n donde la n es el número de versión deseado por el usuario.



```

naam@naam-desktop: ~
naam@naam-desktop:~$ python --version
Python 2.7.17
naam@naam-desktop:~$ python3 --version
Python 3.6.9
naam@naam-desktop:~$ python3.8 --version
Python 3.8.13
naam@naam-desktop:~$ 

```

Figura 48. Versiones instaladas de Python en Jetson nano. Elaboración propia.

3.6.8. Instalación de un IDE en el ordenador de placa y creación de entorno virtual

El IDE más utilizado a nivel global en cualquier sistema operativo incluyendo Ubuntu, es visual studio code. Por lo que se puede descargar sencillamente desde la página oficial seleccionando la arquitectura correspondiente de la placa.

El proceso de instalación es común y corriente, se puede obtener el siguiente resultado:

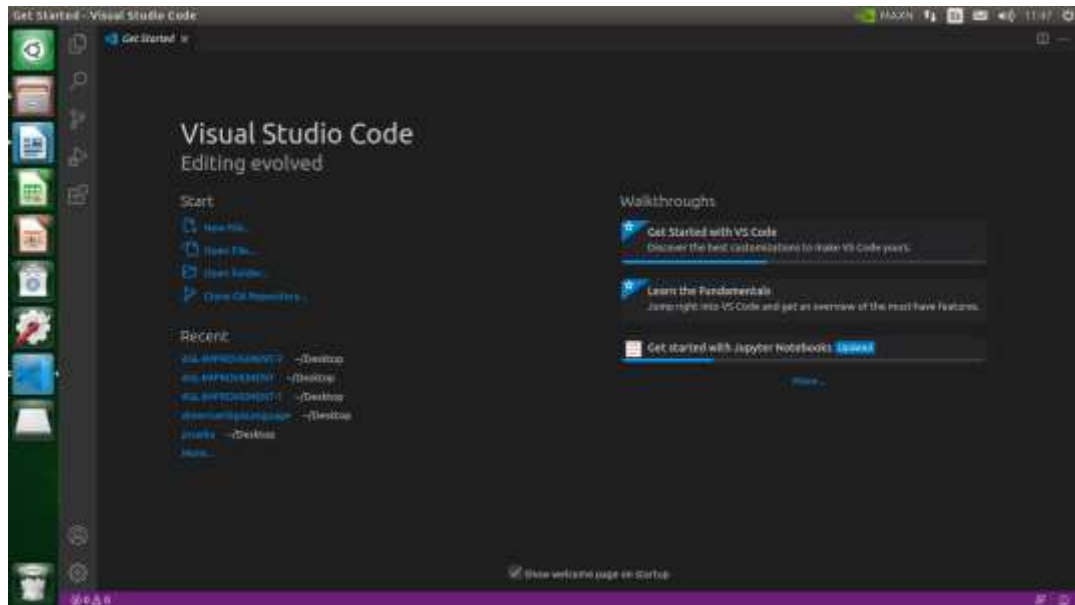


Figura 49. Visual Studio Code en Jetson Nano. Elaboración propia.

Luego de eso es necesario instalar la extensión de Python y de ambientes virtuales para complementar a este popular IDE.

Una vez instaladas las extensiones se procede a hacer uso de los ambientes virtuales de la siguiente manera que es al escribir en la terminal de visual studio code, `virtualenv` y seguido del nombre del ambiente virtual que se va a desear. En este caso queda de la siguiente manera.

Para activar el ambiente se utiliza `source` seguido de `nombredelambiente/bin/activate` y para desactivarlo en la consola se escribe solamente “`deactivate`”. Para información más detallada verficar el Anexo N° 3.

3.6.9. Testeo y predicción en tiempo real

Para el testeo inicial, es necesario entrar en el IDE de confianza de la placa ordenador, como se mostró anteriormente será visual studio code por su disponibilidad para el sistema en el que se trabaja y para posterior a ello activar un ambiente virtual mediante el comando `source Entornovirtual/bin/actívale`. Donde `Entornovirtual` corresponde al nombre del ambiente virtual que el usuario va a otorgar como se muestra en el ANEXO N° 3.



Figura 50. Activación de entorno virtual en el prototipo. Elaboración propia.

Una vez activado el ambiente virtual es necesario la instalación de librerías que se utilizaran y las versiones también va a influir en la compilación del programa. En la tabla 8 se muestra las librerías necesarias y sus versiones para la ejecución del programa:

Tabla 15. "Librerías y versiones específicas para el sistema".

Librerías	Versiones
Tensorflow	2.3 o superior
Mediapipe	0.8.1
CVzone	1.5.5
Opencv	3.4.2 o superior
Scikit	0.23.2

Información proveniente de la investigación de la propia tesis. Elaborado por Alarcón Nelson.

Para la instalación de las librerías se debe ir a la terminal del IDE y mediante el comando pip install seguido del nombre de la librería, comenzará la instalación de las mismas. En caso de que se desee una versión anterior a la descargada, lo que se debe hacer es especificarla mediante dos signos igual y el número de versión requerido como por ejemplo con la instalación de Tensorflow:

- Pip install Tensorflow == 2.3.2

Una vez realizado la instalación de versiones correctas de las librerías a utilizar, se procede a ejecutar el archivo principal desde la terminal con el comando Python seguido del nombre del archivo que este caso es main, debido a que es el nombre que viene por defecto. En la figura 51 se puede observar cómo se ejecutó correctamente el programa.

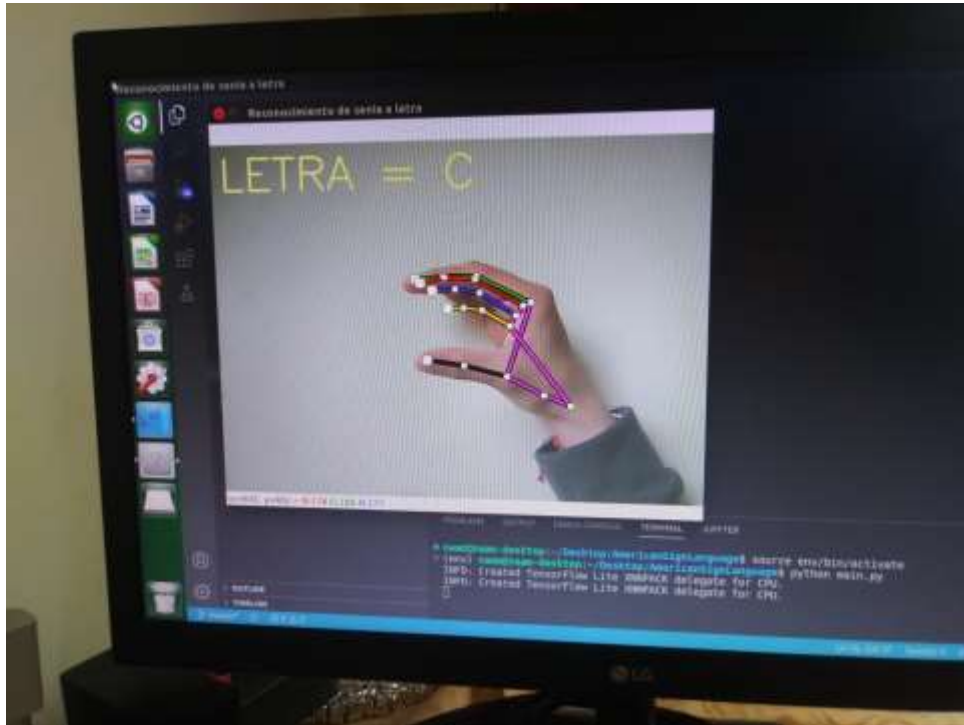


Figura 51. Compilación del traductor de señas en placa Jetson Nano. Elaboración propia.

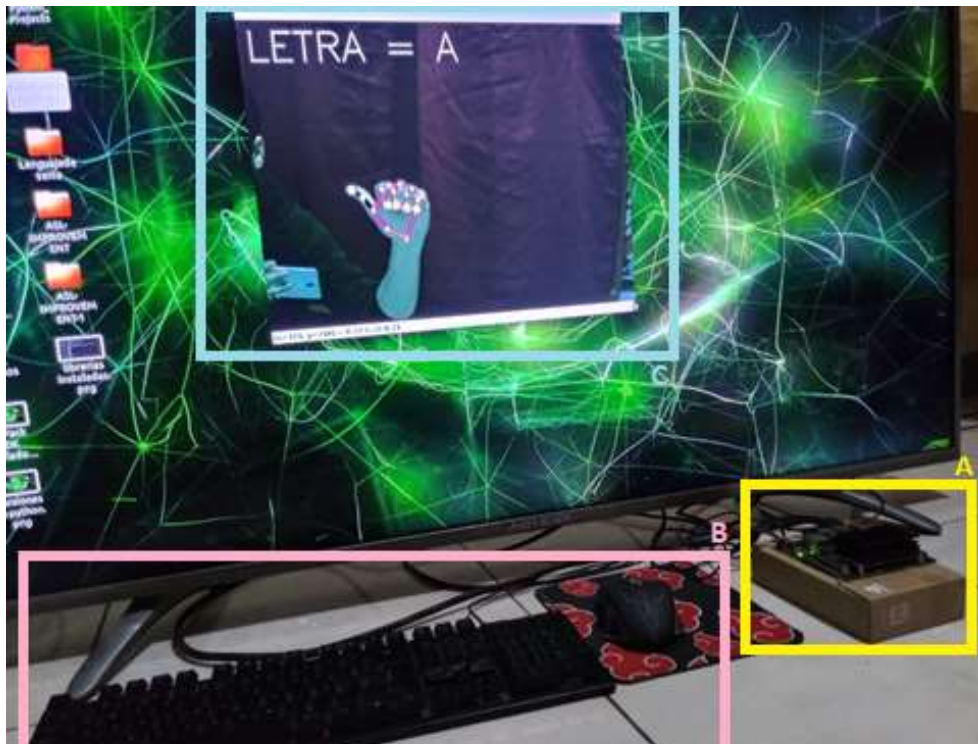


Figura 52. Prototipo traductor de señas en funcionamiento. Elaboración propia.

En la imagen se puede observar de manera segmentada todo el prototipo en funcionamiento en donde:

- A. Corresponde a la Jetson Nano developer kit encendida con una fuente de alimentación de tipo micro-usb.
- B. Periféricos de entrada y salida conectados para controlar el prototipo.
- C. Sistema funcionando en un monitor.

3.7 Implementación del prototipo

Para la implementación de este prototipo se utilizó como objetivo un restaurante de comidas rápidas, debido a que lo que se desea tratar es la inclusión de personas con algún tipo de discapacidad auditiva en lugares donde los tiempos de respuestas son bajos y la afluencia es mayor. En este caso el lugar a elección ha sido McDonald's de un centro comercial.

Mediante la observación se registró que los tiempos de espera normalmente con una persona con esta discapacidad, superan los 3:30 minutos, lo cual las políticas internas del restaurante indican que la entrega final del producto y despedida debe ser dentro de los 3 minutos desde que se forma en una fila el potencial cliente.

A medida que el producto contenga más agregados va a ser mayor el tiempo de espera. Sin embargo, con el prototipo implementado, gracias al deletreo de las letras los cajeros más experimentados y con conocimiento mayor en el menú de alimentos, lograban terminar la orden con las primeras letras proporcionadas por el diseño del traductor. Lo cual hacía que los tiempos de espera logren estar por debajo del límite de 3 minutos.

Para los cajeros pocos experimentados o nuevos en el área debían esperar a que la persona en cuestión terminara de deletrear toda la orden o gran parte de la misma. En donde dependiendo de la velocidad con que se deletree y el tamaño de la orden se podía mantener entre lapsos de tiempo de 2:30 minutos hasta los 4:00 donde se entrega la orden. Evidencia de este proceso queda adjunto en el ANEXO N° 4 y para observar detalladamente el prototipo y los periféricos que lo acompañan consultar ANEXO N° 8.

3.8 Escenarios de pruebas

Como se ha especificado a lo largo del desarrollo de este prototipo, existen escenarios en los que el prototipo puede funcionar de manera adecuada y escenarios en los que el prototipo

puede arrojar fallas considerables. En este inciso se detallan brevemente estos escenarios encontrados al momento.

Pero se debe recalcar que en este apartado se va a especificar el escenario idóneo para el funcionamiento del prototipo en un solo inciso y los incisos restantes son para describir que condiciones pueden afectar al correcto funcionamiento del mismo.

Para ser exactos existe un escenario ideal con dos condiciones y aproximadamente 5 escenarios en los que varias condiciones por el mismo ambiente poco controlado o el usuario va a dar como resultado una predicción poco factible.

El primero de todos ellos lo vamos a denotar como “Escenario idóneo” donde influye directamente la iluminación, distancia y fondo detrás del usuario.

En los otros incisos se va a detallar uno a uno cualquier condición o característica que va a dar como resultado una mala predicción.

3.8.1 Escenario idóneo

En este escenario está compuesto por un fondo monocromático, sin mucho ruido alrededor. Una iluminación adecuada y finalmente la persona que realiza la seña tiene una mano de tamaño promedio al igual que cuenta con todos sus dedos. En la primera imagen de este inciso se muestra como la seña realizada por el usuario que corresponde a la letra A y el programa muestra la misma letra.



Figura 53. Prueba del traductor en escenario ideal. Elaboración propia.

Mientras que en la segunda imagen se puede observar en conjunto con el prototipo en funcionamiento y a su vez ejecutar el programa. Donde se realiza la seña B y el programa en cuestión devuelve el mismo resultado.

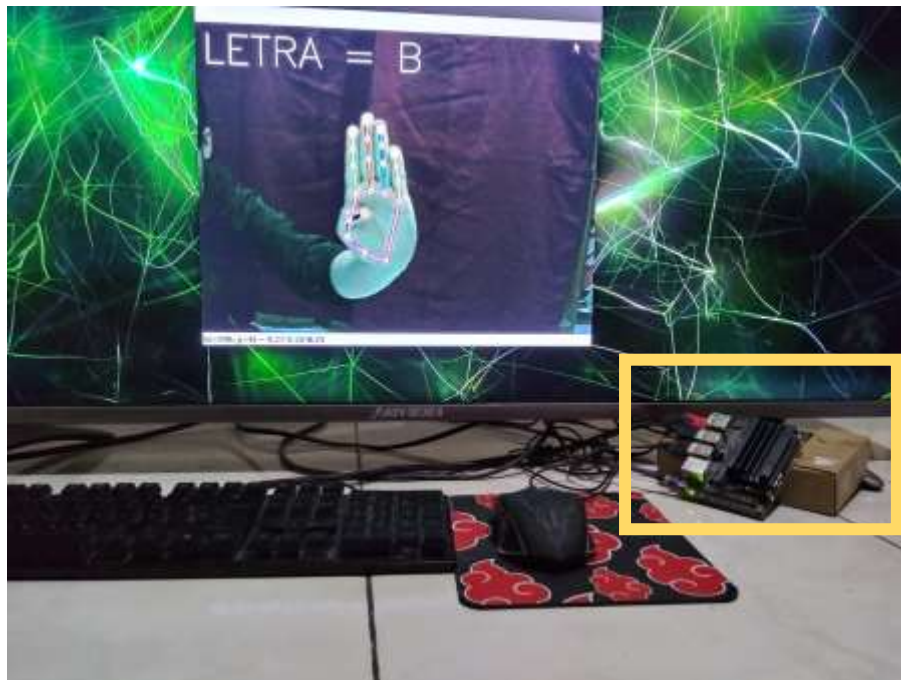


Figura 54. Prototipo funcionando en escenario ideal. Elaboración propia.

3.8.2 Escenario con poca iluminación

En este escenario si no se tiene mucha cercanía a la cámara el programa va a confundir la seña realizada y el sistema los reconoce como otra letra, dando como resultado la mala traducción de la seña realizada por el usuario, en la primera imagen del inciso de escenario con poca iluminación se muestra como resultado la letra P, sin embargo, la seña realizada corresponde a la letra A.



Figura 55. Prueba del traductor de señas en escenario con poca iluminación. Elaboración propia.

En la segunda imagen de este inciso se observa cómo ni siquiera el programa reconoce la mano para fijar los 21 puntos de referencia a causa de la escasa iluminación.

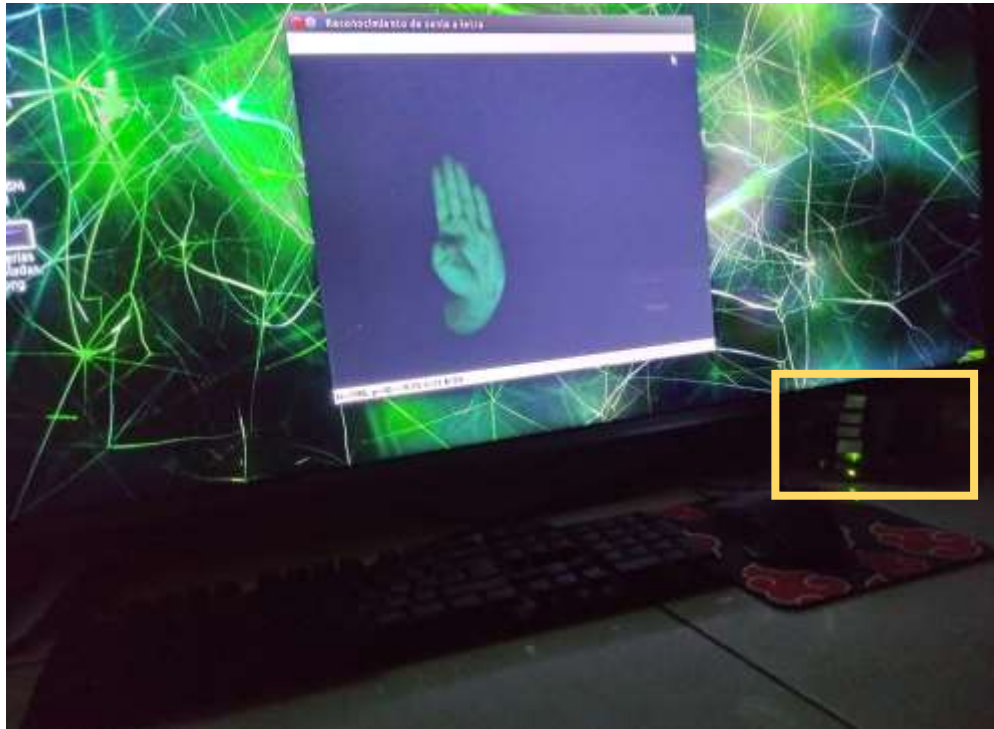


Figura 56. Prototipo funcionando en escenario con poca iluminación. Elaboración propia.

3.8.3 Escenario con mucho ruido

Cuando no se tiene un fondo monocromático, sino que por el contrario existe mucho ruido y objetos en el rango para ejecución de señas, la cámara no logrará asignar los 21 puntos de referencias y por consecuente la seña no podrá traducirse de manera óptima. Se observa en la primera imagen del inciso escenario con ruido como la seña realizada por el usuario corresponde a la letra B y la que refleja el sistema es la letra C. El programa fue ejecutado en el prototipo y solo se capturó el resultado ya compilado.

Por ruido también vamos a considerar cualquier objeto irregular, sobre todo abundancia de objetos de distintas formas y colores.

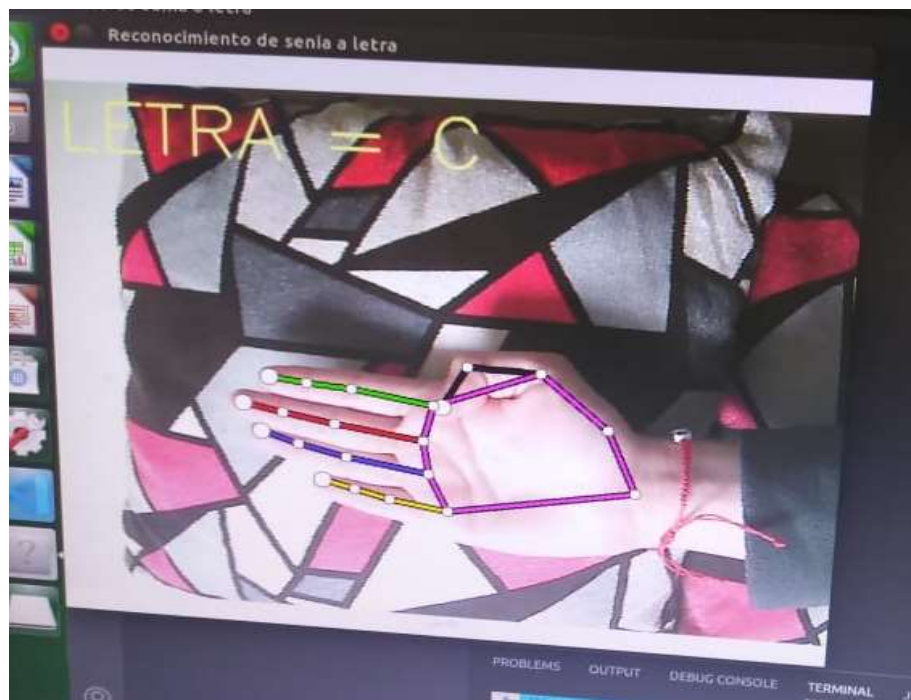


Figura 57. Prueba del traductor en escenario con mucho ruido. Elaboración propia.

En la segunda imagen de este inciso se puede notar como por varios elementos logra producir que el programa no pueda reconocer la palma correctamente. Por lo que da lugar a error en la traducción ya que se realiza la seña correspondiente a la letra A y el prototipo arroja la Q.

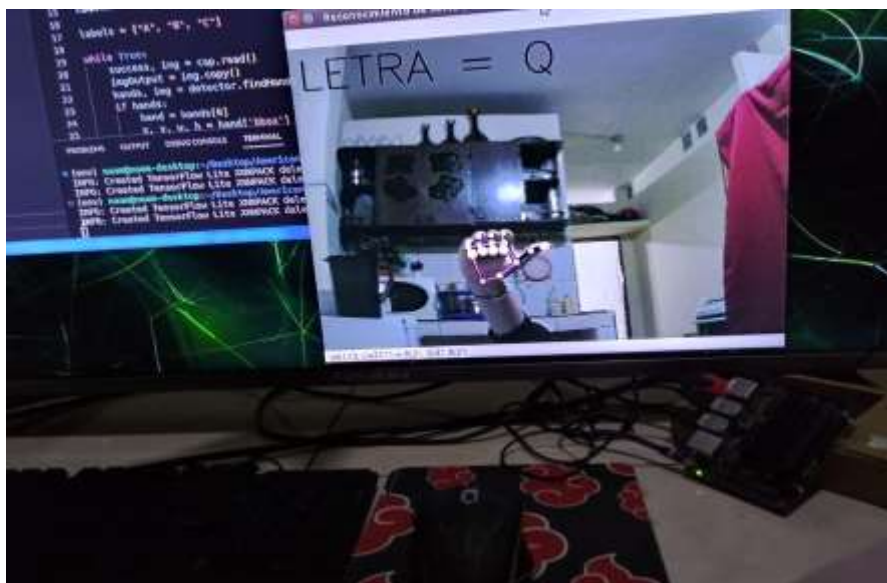


Figura 58. Prototipo funcionando en escenario con poco ruido. Elaboración propia.

3.8.4 Escenario usuario se encuentra lejos

Cuando la mano del usuario se encuentra muy distante de la cámara el sistema tratará de asignar los 21 puntos de un tamaño promedio a una mano muy diminuta lo que hará que estos puntos se confundan entre sí y ocasionando que la seña nunca sea traducida. Esto también se puede ver reflejado en la siguiente imagen donde la seña realizada corresponde a la letra A y el sistema refleja la letra P.



Figura 59. Prueba del traductor en escenario en que el usuario se encuentra lejos. Elaboración propia.

En la segunda imagen se muestra en el programa que el usuario realiza la seña correspondiente a la letra A pero no se refleja respuesta.

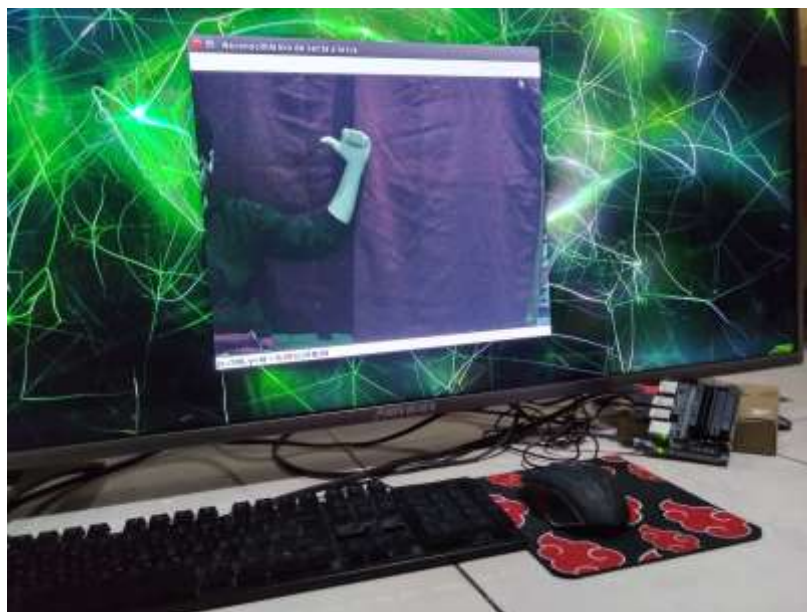


Figura 60. Prototipo funcionando en escenario con mucho ruido. Elaboración propia.

3.8.5 Escenario donde el usuario no cuenta con su extremidad completa

Es evidente que en este escenario es imposible la traducción correcta de señas, la explicación para ello es que se hace uso de la librería mediapipe y de las funciones donde para rastrear las manos la librería hace uso de 21 puntos de referencia los cuales posiciona en toda la mano, desde la palma hasta cada uno de los dedos, por lo que si alguna parte de la mano llega a faltar no podría dibujar los 21 puntos lo que da como resultado una mala traducción o de plano no se traduce.

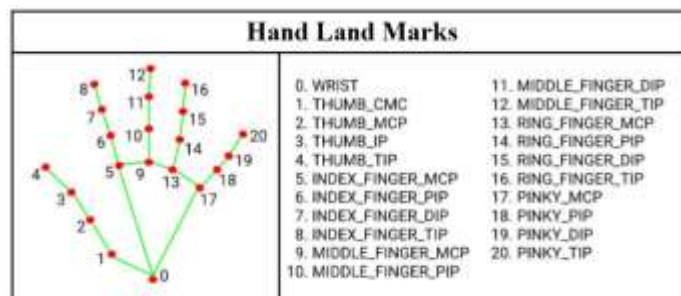


Figura 61. Puntos de referencia de mediapipe. Obtenido de <https://www.geeksforgeeks.org/face-and-hand-landmarks-detection-using-python-mediapipe-opencv/>

3.8.6 Escenario donde el usuario usa un guante

En este escenario donde el usuario utiliza un guante existe error al momento de traducir la seña debido a que Mediapipe que es la librería encargada de detectar y seguir la mano, sin embargo, su forma de operar es que mediante un modelo de IA creado por el equipo de Google permite la detección de la palma de la mano, con el fin de ser más óptimo y rápido, no como el caso de librerías antiguas donde era un reconocimiento de la mano entera y su forma, pero no era tan rápido. Por lo que una vez detectada la palma se procede a destinar los 21 puntos de referencia para la mano. Por lo que al utilizar un guante no reconocerá la palma y por consiguiente la mano haciendo imposible la traducción de seña.

Conclusiones

1. Para la creación de un prototipo orientado a la inteligencia o visión artificial, la opción más completa y eficaz como micro ordenador u ordenador de placa es la Nvidia Jetson Nano debido a la GPU dedicada con la que cuenta.
2. OpenCV y Mediapipe cuentan con una gran documentación a la vez de que está disponible para varios lenguajes de programación. Lo que da como resultado varias funciones dedicadas a la visión artificial y que su implementación en este tipo de

proyectos/sistemas sea sencillo y altamente eficaz en comparación a métodos de la década pasada.

3. Con el prototipo se alcanzaron tiempos de esperas adecuados como los que un cliente regular suele hacer en sus esperas. Fomentando así una inclusión, comodidad y menor dependencia de personas con discapacidad auditiva.
4. Se puede crear una base de datos personalizada, no solo limitarla a letras, pero los mismos deben ser estáticos y no dinámicos.
5. Python resulta ser un lenguaje ideal para desarrollar este tipo de trabajos debido a que cuenta con varios IDE disponibles y a su vez librerías dedicadas para la IA. Es necesario recalcar que en la mayoría de sus IDE's se puede crear ambientes virtuales por cada proyecto, teniendo únicamente versiones de librerías adecuadas para cada proyecto por independiente.
6. La inspiración de la arquitectura AlexNet es eficiente para comenzar en este tipo de proyectos, pero resulta en errores por no estar demasiado optimizada en algunas ocasiones. Por ejemplo, existen inconvenientes al momento de traducir letras como (N, M, P, R, S, T) debido a su similitud en la seña. Para la creación de una mejor red y más robusta en su predicción es necesario tratar con algoritmos de tratamientos de imágenes a la base de datos, pero esto se deja como una mejora a trabajo futuro.
7. Se necesita de un ambiente controlado y que la persona que realice la seña cuente con la extremidad completa para poder traducir con el prototipo adecuadamente.
8. El utilizar herramientas de software libre dio como resultado la disminución en el presupuesto para el proyecto.

Sin una red neuronal optimizadora no es posible alcanzar más del 80% de precisión en la predicción en tiempo real.

Recomendaciones

1. Para obtener la mayor potencia y que los tiempos de respuesta sean menores en el prototipo, es recomendable que la fuente de alimentación sea del tipo barril ya que al otorgarle mayor energía se harán uso de los 4 núcleos que contiene el procesador. Caso contrario como en el prototipo realizado con una fuente de alimentación del tipo micro usb solo se va a tener activos dos núcleos limitando bastante la potencia del ordenador de placa.
2. Es necesario estar actualizado en librerías ya que cada año en sus nuevas versiones más actualizadas quedan obsoletas funciones importantes. Como lo es en el caso de

Tensorflow que contiene actualizaciones constantes y muchas funciones de años anteriores quedan obsoletas o pueden utilizarse, pero con alternativas poco eficientes revisando la documentación oficial.

3. Resulta altamente eficiente cuando dentro del local de comida rápida hay personal altamente experimentado que conoce el menú, pues de esta manera el prototipo mediante la traducción de señas orientado al deletreo hace que el personal de atención capte y entienda lo que el cliente trata de ordenar.
4. Para la creación de dataset y la predicción es aconsejable que las capturas de imágenes sean en un ambiente con iluminación adecuada y fondo monocromático.
5. Es necesario la creación de entornos virtuales para el desarrollo de este trabajo, debido a la variedad de librerías existentes y sus versiones. Existen muchas alternativas para desarrollar un sistema de este tipo por lo que algunas versiones funcionarán bien, otras no e incluso hay casos en los que se debe ir cambiando de versiones en plena ejecución del código como lo es jupyter notebook. Por lo que se recomienda que por cada proyecto tener un ambiente con librerías específicas para que no llegue a afectar al sistema y los proyectos en general.
6. La RNC debe ser optimizada para mayor eficiencia. Sobre todo, al tratar imágenes que sean lo más pequeñas posibles, de preferencia 200x200 pixeles ya que así podrá ser más rápido la traducción del sistema.
7. Se recomienda para un futuro proyecto la base de este desarrollo para poder incluir:
 - a. La lengua de señas, pero esta vez por palabras y no solo limitado a letras.
 - b. Varias opciones adicionales como la de deletreo y formación de palabras.
 - c. La creación de una interfaz amigable ayudaría bastante para que el usuario comprenda mejor el funcionamiento del prototipo cuando existan más de una opción a utilizar (Deletreo de abecedario, Traducción de palabras, Formación de palabras).
8. Para posibles mejoras en trabajos a futuro, se recomienda continuar con los estándares de programación establecidos a lo largo de este trabajo para que se mantengan las funciones, variables, objetos, entre otros.

ANEXOS

Anexo N° 1

Programación en Python para recolección de data (imágenes)

```
import cv2
import mediapipe as mp
import os

#----- Creamos la carpeta donde almacenaremos el entrenamiento -----
nombre = 'Letra_B'
direccion = 'C:/LENGUAJE-SEÑAS/Fotos/Entrenamiento'
carpeta = direccion + '/' + nombre
if not os.path.exists(carpeta):
    print('Carpeta creada: ' + carpeta)
    os.makedirs(carpeta)

#Asignamos un contador para el nombre de la fotos
cont = 0

#Leemos la camara
cap = cv2.VideoCapture(0)
```

Figura 62. Código de recolección de data #1. Elaboración propia.

```
#-----Creamos un objeto que va almacenar la deteccion y el seguimiento de las manos-----
clase_manos = mp.solutions.hands
manos = clase_manos.Hands() #Primer parametro, FALSE para que no haga la deteccion 24/7
                                #Solo hara deteccion cuando hay una confianza alta
                                #Segundo parametro: numero maximo de manos
                                #Tercer parametro: confianza minima de deteccion
                                #Cuarto parametro: confianza minima de seguimiento

#-----Metodo para dibujar las manos-----
dibujo = mp.solutions.drawing_utils #Con este metodo dibujamos 21 puntos criticos de la mano
```

Figura 63. Código de recolección de data #2. Elaboración propia.

```

while (1):
    ret, frame = cap.read()
    color = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    copia = frame.copy()
    resultado = manos.processo(color)
    posiciones = [] # En esta lista vamos a almacenar las coordenadas de los puntos
    #print(resultado.multi_hand_landmarks) #Si queremos ver si existe la detección

    if resultado.multi_hand_landmarks: #Si hay algo en los resultados entramos al if
        for mano in resultado.multi_hand_landmarks: #Buscamos la mano dentro de la lista de manos que nos da el descriptor
            for id, la in enumerate(mano.landmarks): #Vamos a obtener la información de cada mano encontrada por el ID
                #print(id, la) #Como nos entregan decimales (Proporcion de la imagen) debemos pasarlos a pixeles
                alto, ancho, c = frame.shape #Extraemos el ancho y el alto de los fotogramas para multiplicarlos por la proporción
                cpx, cory = int(la.x*ancho), int(la.y*alto) #Extraemos la ubicación de cada punto que pertenece a la mano en coordenadas
                posiciones.append([id, cpx, cory])
                dibujo.draw_landmarks(frame, mano, clase_manos.HAND_CONNECTIONS)

            if len(posiciones) != 0:
                pto_11 = posiciones[4] #5 Dedos: 4 | 0 Dedos: 3 | 1 Dedo: 2 | 2 Dedos: 3 | 3 Dedos: 4 | 4 Dedos: 0
                pto_12 = posiciones[20] #5 Dedos: 20 | 0 Dedos: 17 | 1 Dedo: 17 | 2 Dedos: 20 | 3 Dedos: 20 | 4 Dedos: 20
                pto_13 = posiciones[12] #5 Dedos: 12 | 0 Dedos: 10 | 1 Dedo: 10 | 2 Dedos: 10 | 3 Dedos: 12 | 4 Dedos: 12
                pto_14 = posiciones[0] #5 Dedos: 0 | 0 Dedos: 0 | 1 Dedo: 0 | 2 Dedos: 0 | 3 Dedos: 0 | 4 Dedos: 0
                pto_15 = posiciones[9] #Punto central
                x1, y1 = (pto_15[1]-100), (pto_15[2]-100) #Obtenemos el punto inicial y las longitudes
                ancho, alto = (x1+200), (y1+200)
                x2, y2 = x1 + ancho, y1 + alto
                dedos_reg = copia[y1:y2, x1:x2]
                cv2.rectangle(frame, (x1, y1), (x2, y2), (0, 255, 0), 3)
                dedos_reg = cv2.resize(dedos_reg, (200, 200), interpolation=cv2.INTER_CUBIC) #Redimensionamos las fotos
                cv2.imwrite(carpetas + "/dedos-{}.jpg".format(cont), dedos_reg)
                cont = cont + 1

    cv2.imshow("Video", frame)
    k = cv2.waitKey(1)
    if k == 27 or cont == 100:
        break
cap.release()

```

Figura 64. Código de recolección de data #3. Elaboración propia.

Anexo N° 2

Programación en Python sobre la creación de la Red Neuronal Convolutiva para entrenamiento y creación de modelo y pesos.

Iniciar declarando todas las librerías y funciones a utilizar, adicional a ello hay que dejar establecido el path o ruta donde se encuentra la data recolectada. Recordando que en el código de recolección de data se creó carpetas de entrenamiento (Contiene 80% de las data total) y de validación (Contiene 20% de la data total).

```
#----- Importamos librerías -----
import tensorflow.keras.optimizers

#----- Crean modelo y entrenarlo -----
from tensorflow.python.keras.preprocessing.image import ImageDataGenerator #Nos ayuda a preprocesar las imagenes que le entregamos al modelo
from tensorflow.python.keras import optimizers #Optimizador con el que vamos a entrenar el modelo
from tensorflow.python.keras.models import Sequential #Nos permite hacer redes neuronales secuenciales
from tensorflow.python.keras.layers import Dropout, Flatten, Dense, Activation #
from tensorflow.python.keras.layers import Convolution2D, MaxPooling2D #Capas para hacer las convoluciones
from tensorflow.python.keras import backend as K #Si hay una sesion de keras, lo cerramos para tener todo limpio

K.clear_session() #Limpiamos todo

datos_entrenamiento = 'C:/LENGUAJE-SENAS/Fotos/Entrenamiento'
datos_validacion = 'C:/LENGUAJE-SENAS/Fotos/Validacion'
```

Figura 65. Librerías y funciones de la RNC. Elaboración propia.

El siguiente paso será establecer los parámetros que por defecto se utilizan en una RNC

```
#Parametros
iteraciones = 30 #Numero de iteraciones para ajustar nuestro modelo
altura, longitud = 200, 200 #Tamaño de las imagenes de entrenamiento
batch_size = 1 #Numero de imagenes que vamos a enviar
pasos = 300/1 #Numero de veces que se va a procesar la informacion en cada iteracion
pasos_validacion = 300/1 #Despues de cada iteracion, validamos lo anterior
filtrosconv1 = 32
filtrosconv2 = 64
filtrosconv3 = 128 #Numero de filtros que vamos a aplicar en cada convolucion
filtrosconv4 = 256
filtrosconv5 = 512
tam_filtro1 = (6,6)
tam_filtro2 = (5,5) #Tamaños de los filtros 1 2 3 4 y 5
tam_filtro3 = (4,4)
tam_filtro4 = (3,3)
tam_filtro5 = (2,2)
tam_pool = (2,2) #Tamaño del filtro en max pooling
clases = 26 #26 letras del abecedario American Sign Language
lr = 0.0005 #ajustes de la red neuronal para acercarse a una solucion optima
```

Figura 66. Parámetros establecidos para la RNC. Elaboración propia.

Luego es algo adicional al modelo AlexNet, pero se tomó la decisión de preprocesar las imágenes para otorgar mayor información al sistema. Ya que en este preprocesamiento se busca distorsionar las imágenes existentes para poder acostumbrar el sistema a diversos tipos de manos y que siga siendo altamente eficiente en sus predicciones.

```
#Pre-Procesamiento de las imagenes
preprocesamiento_entre = ImageDataGenerator(
    rescale= 1./255, #Pasar los pixeles de 0 a 255 | 0 a 1
    shear_range = 0.3, #Generar nuestras imagenes inclinadas para un mejor entrenamiento
    zoom_range = 0.3, #Genera imagenes con zoom para un mejor entrenamiento
    horizontal_flip=True #Invierte las imagenes para mejor entrenamiento
)

preprocesamiento_vali = ImageDataGenerator(
    rescale = 1./255
)

imagen_entreno = preprocesamiento_entre.flow_from_directory(
    datos_entrenamiento, #Va a tomar las fotos que ya almacenamos
    target_size = (altura, longitud),
    batch_size = batch_size,
    class_mode = 'categorical', #Clasificacion categorica = por clases
)

imagen_validacion = preprocesamiento_vali.flow_from_directory(
    datos_validacion,
    target_size=(altura, longitud),
    batch_size= batch_size,
    class_mode='categorical'
)
```

Figura 67. Preprocesamiento de imágenes para mayor información. Elaboración propia.

Luego dar por iniciada a la RNC de tipo secuencial y en conjunto con las 5 capas correspondientes de pooling y convoluciones junto a su activación.

```
#Creamos la red neuronal convolucional (CNN)
cnn = Sequential() #Red neuronal secuencial
#Agregamos filtros con el fin de volver nuestra imagen muy profunda pero pequeña
cnn.add(Convolution2D(filtrosconv1, tam_filtro1, padding = 'same', input_shape=(altura, longitud, 3), activation = 'relu')) #Agregamos la primera capa
#Es una convolucion y realizamos config
cnn.add(MaxPooling2D(pool_size=tam_pool)) #Despues de la primera capa vamos a tener una capa de max pooling y asignamos el tamaño
#Maxpooling es la extracción de características

cnn.add(Convolution2D(filtrosconv2, tam_filtro2, padding = 'same', activation='relu')) #Agregamos nueva capa
cnn.add(MaxPooling2D(pool_size=tam_pool))

#Nueva capa 1
cnn.add(Convolution2D(filtrosconv3, tam_filtro3, padding = 'same', activation='relu')) #Agregamos nueva capa
cnn.add(MaxPooling2D(pool_size=tam_pool))

#Nueva capa 2
cnn.add(Convolution2D(filtrosconv4, tam_filtro4, padding = 'same', activation='relu')) #Agregamos nueva capa
cnn.add(MaxPooling2D(pool_size=tam_pool))

#Nueva capa 3
cnn.add(Convolution2D(filtrosconv5, tam_filtro5, padding = 'same', activation='relu')) #Agregamos nueva capa
cnn.add(MaxPooling2D(pool_size=tam_pool))
```

Figura 68. Inicio de la RNC y sus capas ocultas. Elaboración propia.

Finalmente se tiene el aplanamiento para ser más rápida la red, el entrenamiento y la creación de los modelos que se van a usar en las predicciones.

```
#Ahora vamos a convertir esa imagen profunda a una plana, para tener 1 dimension con toda la info
cnn.add(Flatten()) #Aplanamos la imagen
cnn.add(Dense(4096, activation='relu')) #Asignamos 4096 neuronas
cnn.add(Dropout(0.5)) #Apagamos el 50% de las neuronas en la funcion anterior para no sobreajustar la red
cnn.add(Dense(clases, activation='softmax')) #Es nuestra ultima capa, es la que nos dice la probabilidad de que sea alguna de las clases

#Agregamos parametros para optimizar el modelo
#Durante el entrenamiento tenga una autoevaluacion, que se optimice con Adam, y la metrica sera accuracy
optimizar = tensorflow.keras.optimizers.Adam(learning_rate=lr)
cnn.compile(loss = 'categorical_crossentropy', optimizer= optimizar, metrics=['accuracy'])

#Entrenaremos nuestra red
cnn.fit(imagen_entreno, steps_per_epoch=pasos, epochs= iteraciones, validation_data= imagen_validacion, validation_steps=pasos_validacion)

#Guardamos el modelo
cnn.save('Modelos.h5')
cnn.save_weights('Pesos.h5')
```

Figura 69. Entrenamiento de la RNC y su creación de modelos. Elaboración propia.

Anexo N° 3

Instalación de librerías y creación de entorno virtual en placa Jetson Nano mediante Visual Studio Code.

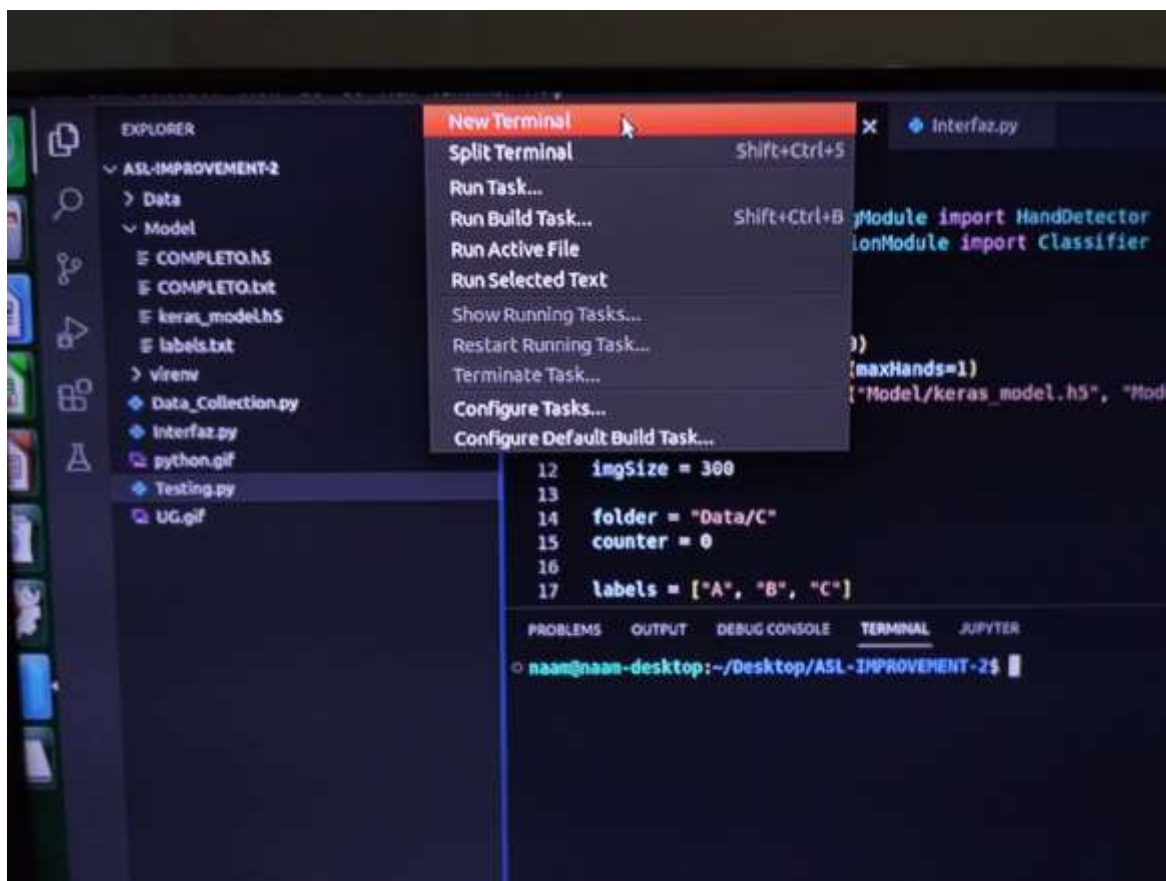


Figura 70. Creación de entorno virtual Entornodeprueba #1. Elaboración propia.

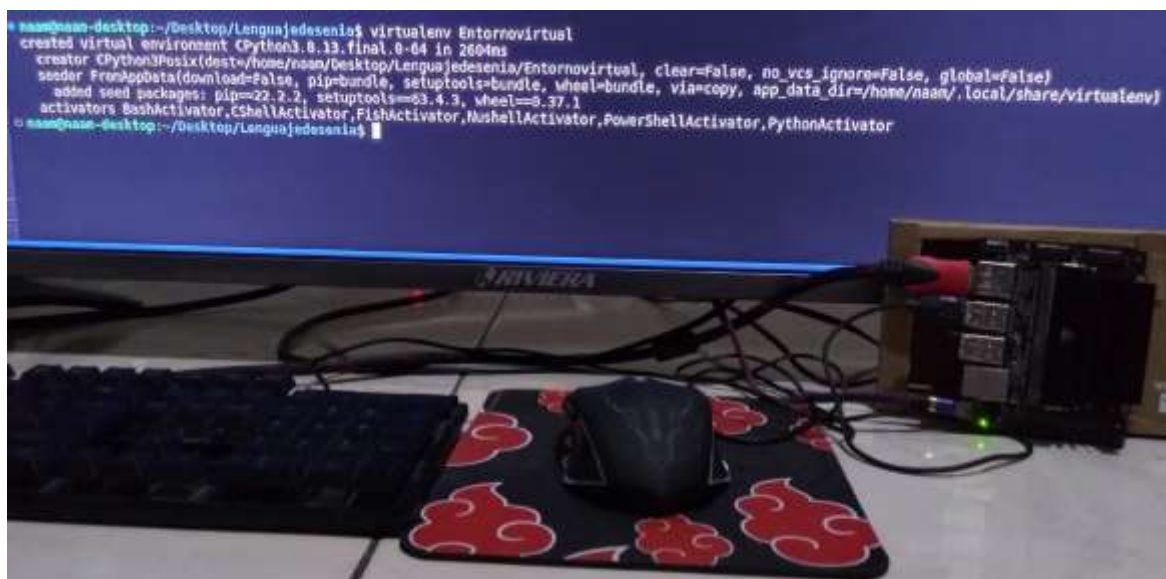
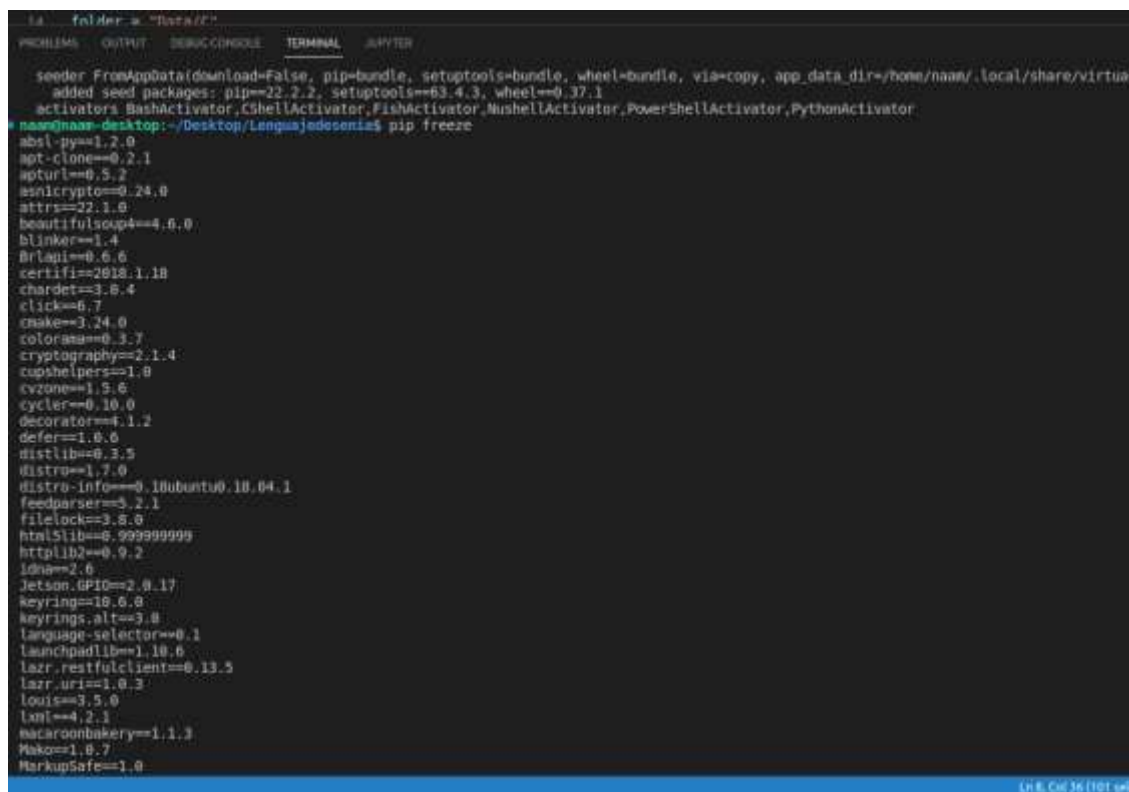


Figura 71. Creación de entorno virtual Entornodeprueba #2. Elaboración propia

Mediante el comando pip freeze se puede observar las librerías instaladas ya sea en el sistema o en el entorno virtual.



```

folder: a: "Data/1/"
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER
seeder FromAppdata(download=False, pip=bundle, setuptools=bundle, wheel=bundle, via=copy, app_data_dir=/home/naam/.local/share/virtual
added seed packages: pip==22.2.2, setuptools==63.4.3, wheel==0.37.1
activators BashActivator, CShellActivator, FishActivator, NushellActivator, PowerShellActivator, PythonActivator
naam@naam-desktop:~/Desktop/Lenguajedesenia$ pip freeze
absl-py==1.2.0
apt==0.2.1
apturl==0.5.2
asn1crypto==0.24.0
attrs==22.1.0
beautifulsoup4==4.6.0
blinker==1.4
Brlapi==0.6.6
certifi==2018.11.18
chardet==3.6.4
click==6.7
cmake==3.24.0
colorama==0.3.7
cryptography==2.1.4
cupshelpers==1.0
cvzone==1.5.6
cycler==0.10.0
decorator==4.1.2
defer==1.0.0
distlib==0.3.5
distro==1.7.0
distro-info==0.18ubuntu0.18.04.1
feedparser==5.2.1
filelock==3.8.0
html5lib==0.999999999
httplib2==0.9.2
idna==2.6
Jetson.GPIO==2.0.17
keyring==18.0.0
keyrings.alt==3.0
language-selector==0.1
launchpadlib==1.10.6
lazr.restfulclient==0.13.5
lazr.uri==1.0.3
louis==3.5.0
lxml==4.2.1
macaroonbakery==1.1.3
Mako==1.0.7
MarkupSafe==1.0

```

Figura 72. Librerías existentes en el prototipo. Elaboración propia.

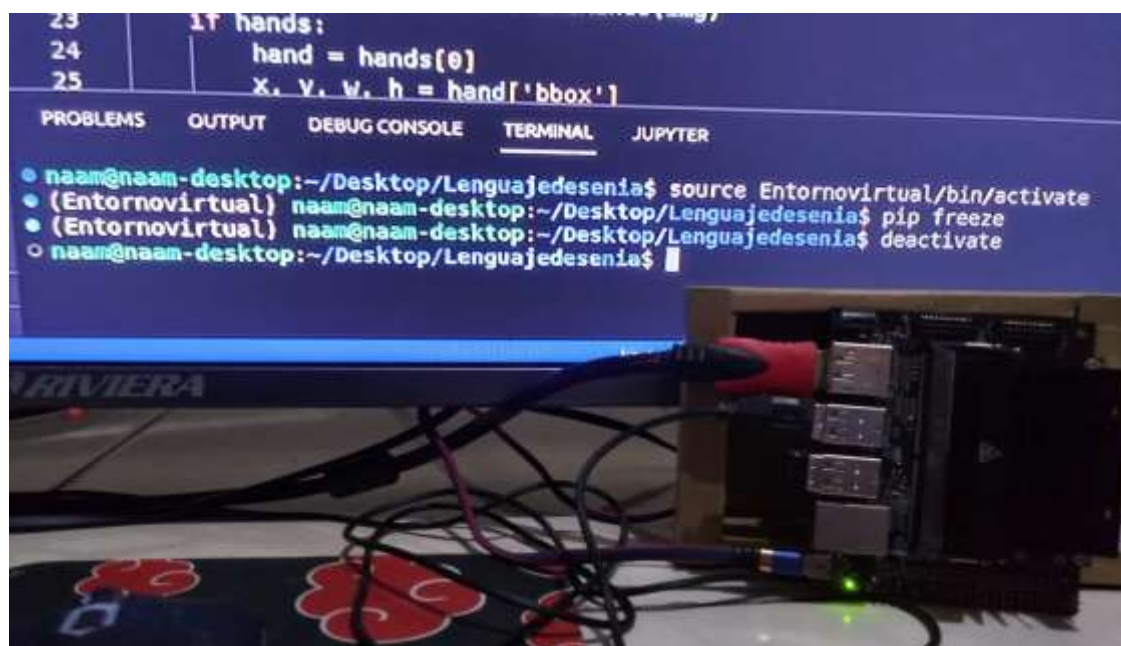


Figura 73. Activación y desactivación del entorno virtual. Elaboración propia.

Anexo N° 4

Pruebas del software traductor de seña a letra en sitios de gran afluencia.

En este anexo se puede observar como el software funciona con normalidad debido a la iluminación y al poco ruido que existía. Se decidió probar en un local de comida rápida pues los tiempos de respuesta van a ser siempre bajos y tienen protocolos de tiempo obligados a cumplir. En la primera y segunda imagen de este anexo se observa que la sea realizada por el usuario corresponde a la letra G.

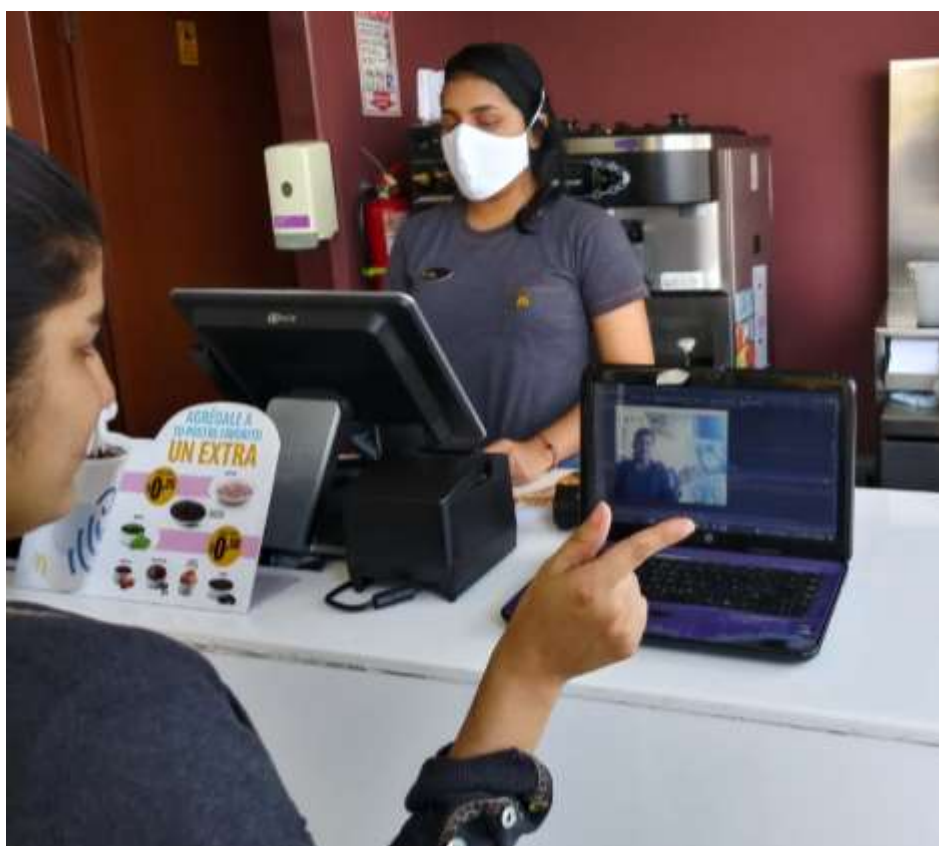


Figura 74. Prueba del programa traductor de seña a texto en lugar con gran afluencia #1. Elaboración propia.

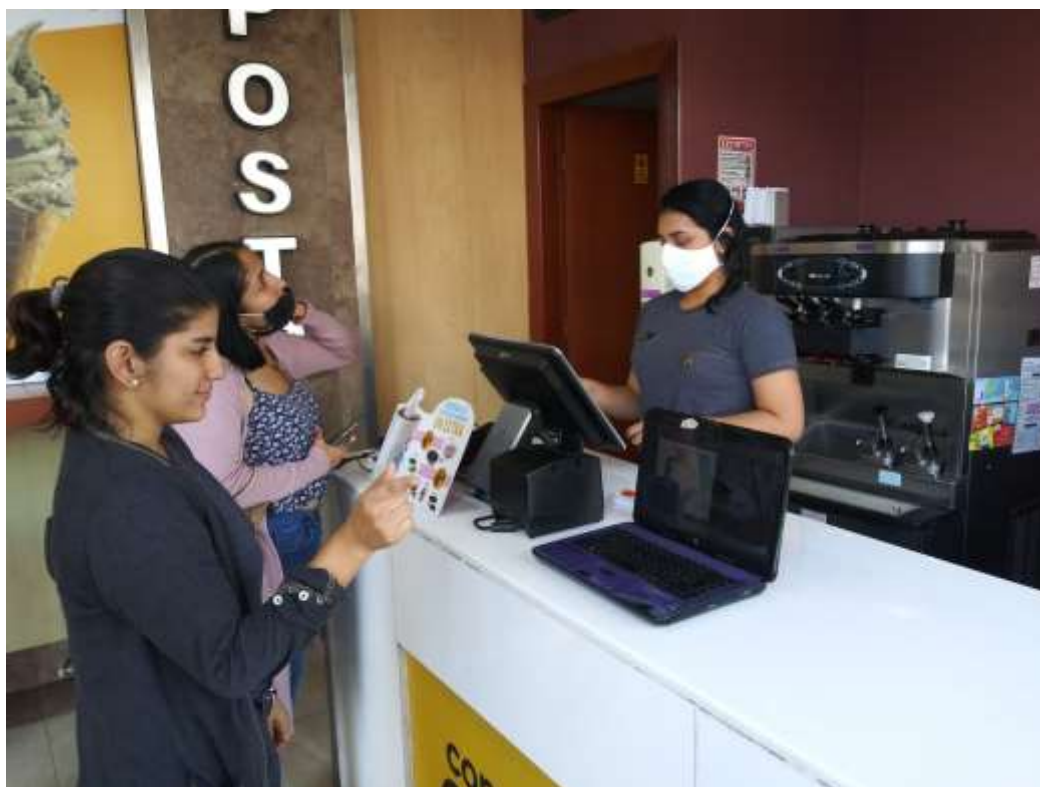


Figura 75. Prueba del programa traductor de seña a texto en lugar con gran afluencia #2. Elaboración propia.

En la tercera imagen de este inciso se observa ya de manera digital como el usuario realiza la seña G y el programa arroja como resultado la G, se puede concluir que efectivamente funciona en un ambiente controlado.



Figura 76. Prueba del programa traductor de seña a texto en lugar con gran afluencia #3. Elaboración propia.

Se puede notar como se procede a probar distintas letras como la B la cual es la seña que se realiza en la imagen número tal y el sistema arroja la misma letra B.



Figura 77. Prueba del programa traductor de seña a texto en lugar con gran afluencia #4. Elaboración propia.



Figura 78. Prueba del programa traductor de seña a texto en lugar con gran afluencia #5. Elaboración propia.

Anexo N° 5**Prueba del prototipo traductor de seña a letra en un ambiente controlado I.**

Se hicieron pruebas iniciales con distintas manos en un ambiente controlado, en este anexo se observa una mano pequeña de una mujer cuya traducción es altamente eficaz debido a la buena iluminación y fondo de por medio.



Figura 79. Inferencia en tiempo real del prototipo #2. Fuente elaboración propia.

Anexo N° 6**Prueba del prototipo traductor de seña a letra en un ambiente controlado II.**

Para el segundo testeo se utilizó una mano de tamaño promedio, al igual que la anterior se tradujo eficazmente y todo debido al ambiente controlado.



Figura 80. Inferencia en tiempo real del prototipo #2. Fuente elaboración propia.



Figura 81. Inferencia en tiempo real del prototipo #3. Fuente elaboración propia.

Anexo N° 7

Instalación de sistema operativo JetPack SDK en la Jetson Nano Developer Kit.

En el paso 1 se especifica como para iniciar con este proceso, se necesita una SD y la misma se va a formatear con un software especial en formateo de tarjetas SD llamado “SD Card Formatter” que se lo puede descargar desde la página de Nvidia Jetson Nano – primeros pasos.

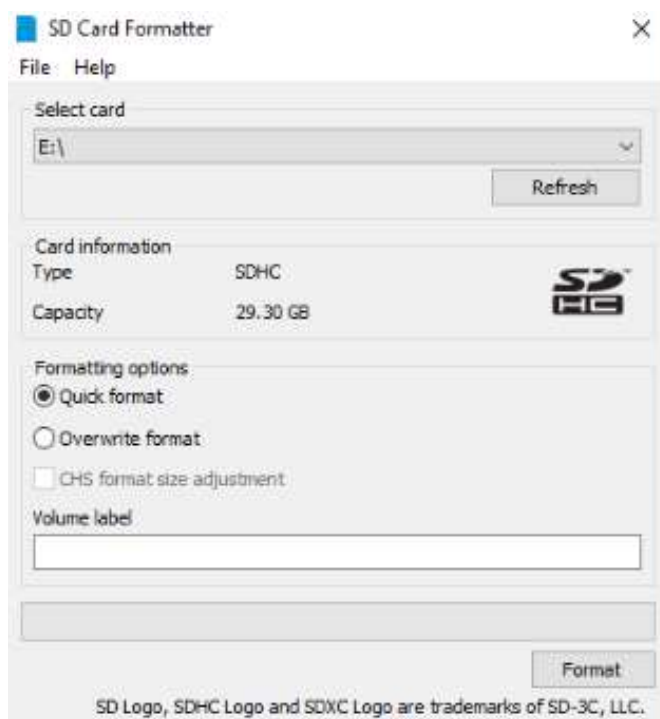


Figura 82. Interfaz de “SD Card Fromatter”. Obtenido de la web.

El segundo paso consta de descargar el software Balena Etcher que está especializado en bootear sistemas operativos de tipo ISO en una tarjeta SD.

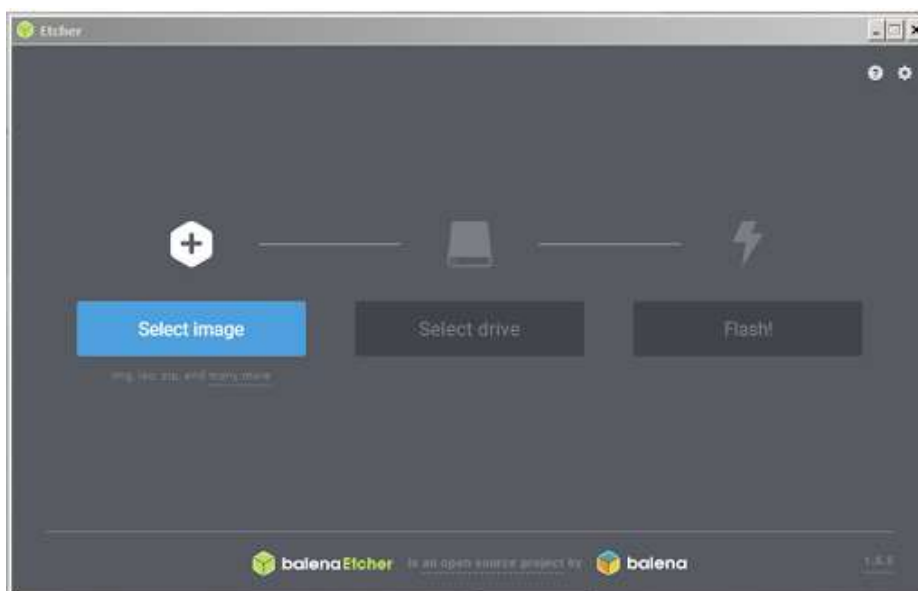


Figura 83. *Interfaz de BalenaEtcher. Obtenido de la web.*

Posterior a ello se debe descargar el software JetPack SDK que la misma organización Nvidia la provee desde su página oficial Jetson Nano – primeros pasos. Una vez se haya descargado el S.O que pesa aproximadamente unos 7GB los cuales dependiendo del internet tendrá un tiempo de espera estimado.

Cuando se tiene el S.O en tipo ISO, la SD conectada al ordenador y Balena Etcher instalado, se puede bootear JetPack SDK al seleccionar la ISO y la SD luego presionar en flash y esperar a que el proceso culmine.

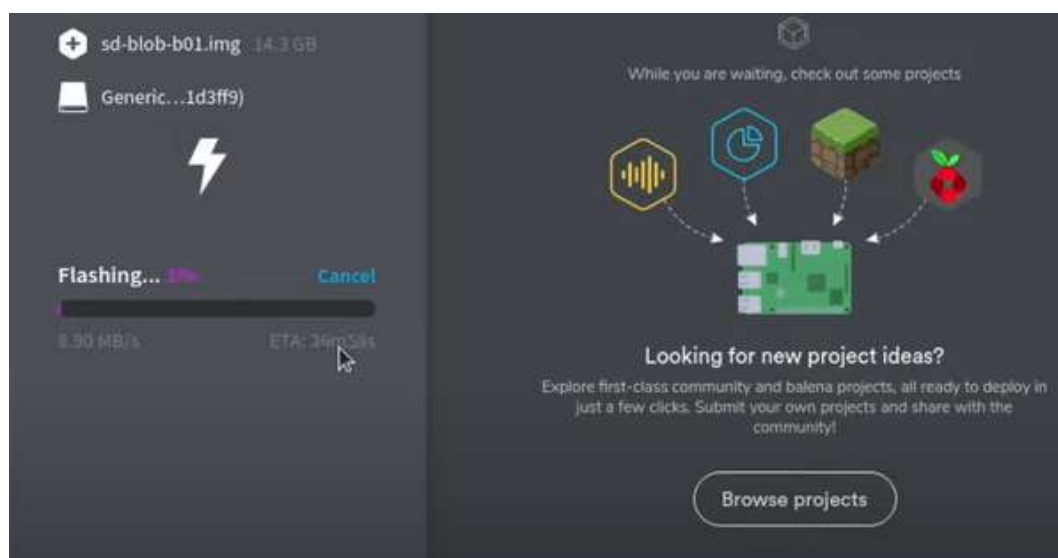


Figura 84. *Flasheo de JetPack SDK en SD. Obtenida de la web. Elaborada por AMP Tech.*

Una vez el sistema se booteo en la SD, se va a insertar la SD en la ranura especial que contiene la JetsonNano para SD. Como se muestra en la siguiente gráfica que proporciona Nvidia.

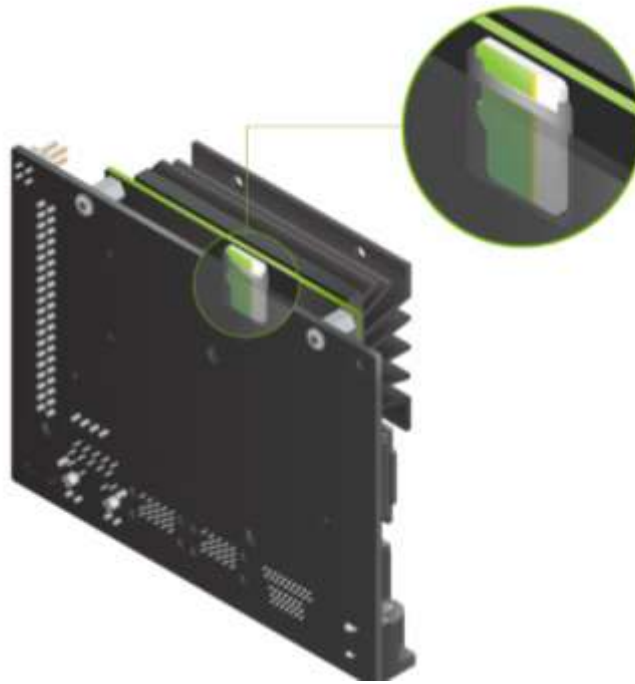


Figura 85. Inserción de SD con Jetpack SDK en Jetson Nano. Obtenido de la web. Elaborado por NVIDIA (2022).

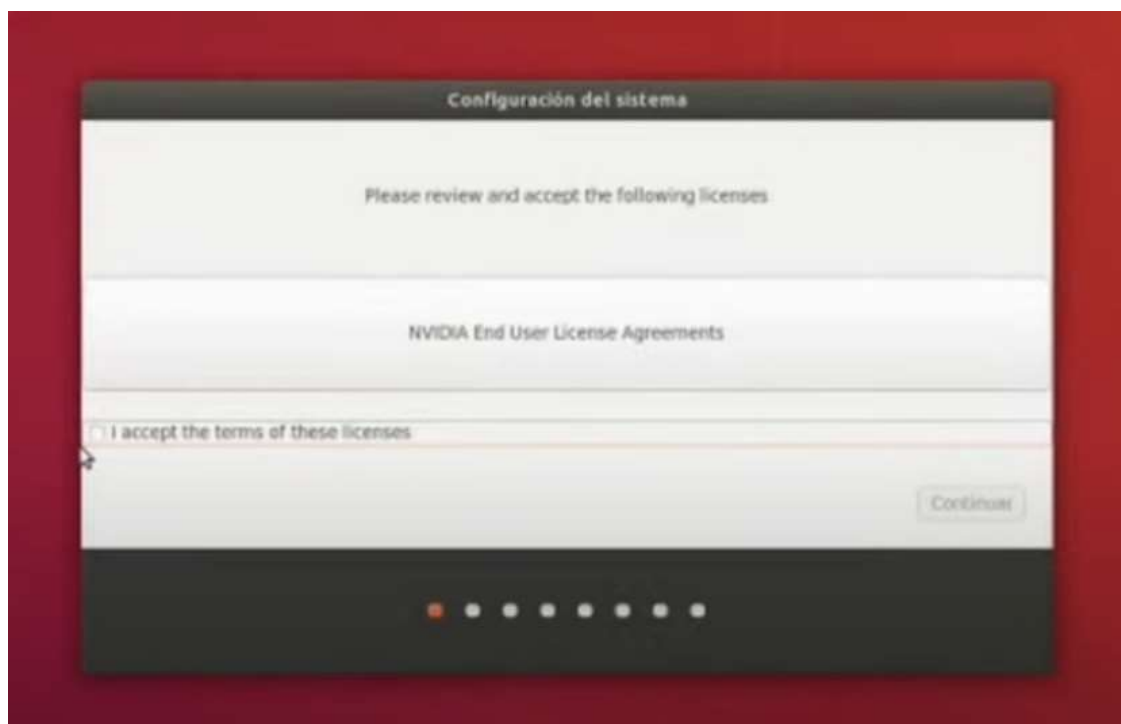


Figura 86. Configuración inicial del S.O. en Jetson Nano. Elaborado por AMP Tech.

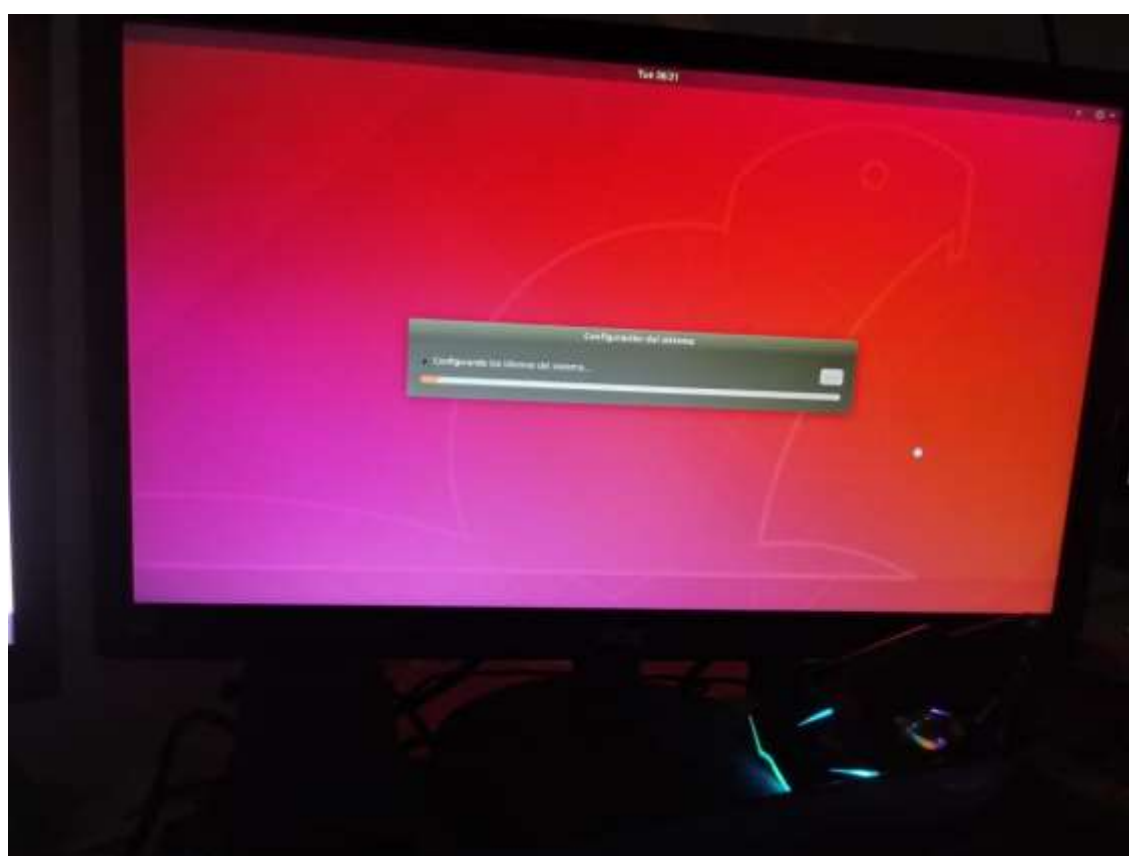


Figura 87. Configuración final del S.O. en Jetson Nano. Elaborado por AMP Tech.



Figura 88. Creación de usuario en JetPack SDK. Elaboración propia.

Una vez realizado la configuración inicial, el sistema se va a reiniciar en algunos casos un par de veces hasta que se muestre el logo de NVIDIA y posterior a se va a tener totalmente instalado el sistema operativo junto con las herramientas CUDA y Python.



Figura 89. Inicio de JetPack SDK en Jetson Nano. Elaboración propia



Figura 90. Interfaz de JetPack SDK. Elaboración propia.

Anexo N° 8

Jetson Nano Developer Kit y sus accesorios



Figura 91. Jetson Nano developer kit y accesorios básicos a utilizar. Elaboración propia.



Figura 92. Jetson Nano con periféricos conectados. Elaboración propia



Figura 93. Jetson Nano developer kit sin periféricos. Elaboración propia



Figura 94. Interfaz digital de JetPack SDK. Elaboración propia.



Figura 95. Prototipo traductor de seña funcionando correctamente. Elaboración propia.

Bibliografía

- Aquino Castro, R. A. (2018). Reconocimiento e interpretación del alfabeto dactilológico de la lengua de señas mediante tecnología móvil y redes neuronales artificiales. (Tesis de grado). Universidad Mayor de San Andrés, La Paz, Bolivia. <http://repositorio.umsa.bo/xmlui/handle/123456789/17474>
- Asamblea Nacional de la República del Ecuador. (2014, 02 de septiembre). Ley Orgánica de discapacidades. (pág. 6, 7 y 9) Enlace: https://www.consejodiscapacidades.gob.ec/wp-content/uploads/downloads/2014/02/ley_organica_discapacidades.pdf
- Berzal, F. (2019). Redes neuronales & deep learning: Volumen ii. Independently published.
- Challenger-Pérez, Ivet; Díaz-Ricardo, Yanet; Becerra-García, Roberto Antonio El lenguaje de programación Python Ciencias Holguín, vol. XX, núm. 2, abril-junio, 2014, pp. 1-13 Centro de Información y Gestión Tecnológica de Santiago de Cuba Holguín, Cuba. Consultado en: <https://www.redalyc.org/pdf/1815/181531232001.pdf>
- Chazallet, S. (2016). Python 3: los fundamentos del lenguaje. Barcelona: Ediciones ENI.
- Chen, H., Liu, S., Zhang, H., y Cheng, W. (2019). Handwriting numerals recognition using convolutional neural network implemented on nvidia's jetson nano. En International conference in communications, signal processing, and systems (pp. 529– 535).
- Chiguano Rodríguez, E. F., & Moreno Díaz, N. V. (16 de Junio de 2011). Escuela Politécnica Nacional. Repositorio: <http://bibdigital.epn.edu.ec/handle/15000/3915>
- Consejo nacional de igualdad y discapacidades (CONADIS). (2014, 24 de Marzo). Sitio Web. Normas Jurídicas en discapacidad Ecuador. (pág. 83-84). Enlace: <https://www.consejodiscapacidades.gob.ec/wp-content/uploads/downloads/2014/08/Libro-Normas-Jur%C3%ADdicas-en-Discapacidad-Ecuador.pdf>

- Cruz-Guerrero, R., & Gutiérrez-Fragoso, K. (2021). Detección de estudiantes que copian en el aula usando Redes Neuronales Convolucionales. *Pädi Boletín Científico de Ciencias Básicas e Ingenierías del ICBI*, 9(Especial), 106-109. DOI: <https://doi.org/10.29057/icbi.v9iEspecial.7492>
- Dominguez, T. (2021). Visión artificial. En T. Dominguez, *Aplicaciones prácticas con OpenCV - Python* (págs. 1-3). MARCOMBO, S. L.
- El Universo. (01 de noviembre de 2021). Estas son las cinco ciudades más pobladas de Ecuador. Quito, Ecuador. <https://www.eluniverso.com/noticias/ecuador/las-cinco-ciudades-mas-pobladas-de-ecuador-nota/>
- Estévez, A. (2022). Desarrollo e implementación de un sistema de identificación de signos de la lengua de señas mexicana basado en redes neuronales y el sistema embebido Jetson Nano (Tesis para obtener el grado de Maestro en Electrónica). Universidad Tecnológica de la Mixteca, Huajuapán de León, Oaxaca. <http://repositorio.utm.mx/handle/123456789/411>
- Fernández Suárez, K. J., & Sandoval Palacios, G. (02 de Octubre de 2017). Diseño y construcción de un prototipo de sistema electrónico para conversión de lenguaje de señas a mensajes de voz para la comunicación de personas sordomudas, en la ciudad de Chiclayo. (Tesis de Ingeniería). Universidad Nacional Pedro Ruiz Gallo. Repositorio Institucional: <https://hdl.handle.net/20.500.12893/1158>
- Free Software Foundation. (28 de junio de 2022). GNU. <https://www.gnu.org/philosophy/free-sw.es.html>
- Granda Cárdenas, A. A. (2020). Comparativa de extractores de características para clasificación de rostros (B.S. tesis). Quito, 2020. Extraído de: <http://bibdigital.epn.edu.ec/handle/15000/20763>
- Gestal, M. (2014). Introducción a las redes de neuronas artificiales. Universidad da Coruña. https://www.researchgate.net/publication/242099672_Introduccion_a_las_Redess_d_e_Neuronas_Artificiales
- Haykin, S. S., y cols. (2009). *Neural networks and learning machines*/simon haykin. New York: Prentice Hall,.

- Herrera, V. (2005). Adquisición temprana de lenguaje de signos y dactilología. *Revista psicopedagógica*, 13(77-78), 2-10. https://www.cultura-sorda.org/wp-content/uploads/2015/03/Herrera_Adquisicion_temprana_LS_y_dactilologia_20051.pdf
- Hirsh-Pasek, K. (1987). The metalinguistic of fingerspelling: An alternate way to increase reading vocabulary in congenitally deaf readers. *Reading research Quarterly*, XXII 4,455-473.
- Howse, . M. J., J. (2020). *Learning openCV 4 computer vision with python 3: Get to grips with tools, techniques, and algorithms for computer vision and machine learning*. Packt Publishing Ltd.
- Instituto Nacional de Estadísticas y Censos. (2021). INEC. <https://www.ecuadorencifras.gob.ec/censo-de-poblacion-y-vivienda/>
- Izaurieta, F., y Saavedra, C. (2000). *Redes neuronales artificiales*. Departamento de Física, Universidad de Concepción Chile. https://d1wqtxts1xzle7.cloudfront.net/36957207/Redes_neuronales-with-cover-page-v2.pdf?Expires=1658792609&Signature=L~L4Iu2ii-s4XW4qgX71p4PbWrN44WqPBEUHI46b1zJGq3jyUMguyEc83hLg21Va3xvj2AE4rEzMz2H3PG0InmXMMELQGJ3IddS7DatoId3yotMoDNdx8~Pnzjmqi2TP38EZw8PpcHCr~Fw~NhoBLCAPMgvDhOM9Y~QVwSOGcwkYpFaSIQ7YcY9thXd5od0sYb4T7T6X92Yr2mN2PB6JEZuX51I89rIHtZNYIOIQNpW07VABZi2z8-3jIhje4jq-4XFdl8~3qS3mvgkophTGMx6YeWGGz6jm~WEv35poKorOEcgO1ZocskrFbKZ5MVO2vvfZQXaxZgGRRrzdmEzICA__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA
- Krizhevsky, A, I Sutskever, y G Hinton (2012), Imagenet classification with deep convolutional neural networks, In *Advances in neural information processing systems*, <https://papers.nips.cc/paper/2012/hash/c399862d3b9d6b76c8436e924a68c45b-Abstract.html>

- Kukil. (01 de Marzo de 2022). LearnOpenCV. <https://learnopencv.com/introduction-to-mediapipe/>
- Lucumi Carvajal, L y Marín Mejía, Á. (2021). Sistema multimedia para la asociación del alfabeto en castellano con el alfabeto dactilológico colombiano para niños sordos de primaria. Universidad Autónoma de Occidente. <https://hdl.handle.net/10614/13550>
- Marcos, A. G., de Pisón Ascacíbar, F. J. M., Elías, F. A., Limas, M. C., Meré, J. B. O., & González, E. P. V. (2006). Técnicas y algoritmos básicos de visión artificial. Técnicas y Algoritmos Básicos de Visión Artificial. <https://investigacion.unirioja.es/documentos/5c13b22ac8914b6ed3778a6a>
- MediaPipe. (2022). MediaPipe.dev. <https://mediapipe.dev/>
- Méndez Gómez, J. (2019). Sistema de reconocimiento facial basado en redes neuronales convencionales sobre el dispositivo Raspberry Pi. <https://idus.us.es/handle/11441/95442>
- Microsoft. (2022). Azure. <https://azure.microsoft.com/es-es/resources/cloud-computing-dictionary/what-is-machine-learning-platform/#benefits>
- Morales, Eduardo & Aparicio, Oswaldo & Arguijo, Pedro & Melendez-Armenta, Roberto & López, José Antonio. (2019). Traducción del lenguaje de señas usando visión por computadora. Research in Computing Science. 148. 79-89. 10.13053/rcs-148-8-6. DOI: 10.13053/rcs-148-8-6
- Mundo Tecnológico (2020, julio 2). ¿Cómo filtrar una imagen en MATLAB? – Curso Procesamiento de imágenes (parte 4). [Archivo de video]. https://www.youtube.com/watch?v=4XaEfczSMaI&ab_channel=MundoTecnol%C3%B3gico
- Nielsen, M. A. (2015). Neural networks and deep learning (Vol. 2018). Determination press San Francisco, CA.
- NVIDIA. (2022). NVIDIA DEVELOPER. <https://developer.nvidia.com/embedded/jetson-nano-developer-kit>
- ORACLE. (2022). Oracle México. <https://www.oracle.com/mx/artificial-intelligence/what-is-ai/>

- Organización Mundial de la Salud (OMS). (2001). Clasificación Internacional de Funcionamiento de la Discapacidad y de la Salud (CIF). Extraído de https://aspace.org/assets/uploads/publicaciones/e74e4-cif_2001.pdf, consulta 08/07/2022
- Padden, C. & Ramsey, C. (2000). American Sign Language and reading ability in deaf children. En C. Chamberlain, J. P. Morford, & R. Mayberry (Eds.), *Acquisition of Language by Eyes*. London. Lawrence Erlbaum Associates.
- Palomera, D. (2015). Aprendizaje semi-supervisado para la detección de respuestas informativas en preguntas procedurales (Tesis Doctoral no publicada). UNIVERSIDAD ANDRES BELLO. <http://repositorio.unab.cl/xmlui/handle/ria/15499>
- Pertusa, E., & Fernández-Viader, M. D. P. (1999). Representación fonológica, aprendizaje de la escritura y alumnos sordos. *Revista de Logopedia, foniatría y audiología*. Consultada de: <http://hdl.handle.net/2445/56927>
- Quirós, J. P. S. (2019). Convolución matricial aplicado al procesamiento de imágenes. <https://www.tec.ac.cr/>.
- Ringa Tech (2021, agosto 16). Redes Neuronales Convolucionales – Clasificación avanzada de imágenes con IA / ML (CNN). [Archivo de video]. https://www.youtube.com/watch?v=4sWhhQwHqug&ab_channel=RingaTech
- Ringa Tech (2021, septiembre 14). Crea un clasificador de perros y gatos con IA, Python y Tensorflow – Proyecto completo. [Archivo de video]. https://www.youtube.com/watch?v=DbwKbsCWPSg&t=655s&ab_channel=RingaTech
- Rivas Asanza, W., & Mazón Olivo, B. (2018). *Redes Neuronales Artificiales Aplicadas al Reconocimiento de Patrones*. Machala - Ecuador: Editorial UTMACH.
- Rojas, E. (07 de Febrero de 2018). <https://www.muycomputerpro.com/2018/02/07/glosario-terminos-basicos-machine-learning>

- Suárez, S. T. P. (2015). Metodologías de diseño de redes neuronales sobre dispositivos digitales programables para procesamiento de señales en tiempo real (Tesis Doctoral no publicada). Universidad de Las Palmas de Gran Canaria.
- Sivisapa, L., & Paz, H. (2016). VISIÓN ARTIFICIAL APLICADA PARA EL RECONOCIMIENTO DEL LENGUAJE DE SEÑAS. GEEKS DECC-REPORTS, 5(1). <https://journal.espe.edu.ec/ojs/index.php/geeks/article/view/271>
- Storch de Gracia y Asensio, José Gabriel (11 de noviembre de 2020) *El nombre de nuestra lengua*. In I Congreso Iberoamericano de Educación Bilingüe para Sordos, 6-10 de julio de 1.998, Lisboa (Portugal). (Presentado). Extraído de: <https://eprints.ucm.es/id/eprint/62622/>
- Universidad en Internet. (29 de Noviembre de 2019). UNIR. <https://www.unir.net/ingenieria/revista/lenguaje-r-big-data/> Universidad Nacional Autónoma de México. (Febrero de 2021). UNAM. <https://docencia.tic.unam.mx/presenciales/Lenguaje-de-programacion-java.html>
- Voisard, A. (23 de Septiembre de 2015). Naciones Unidas. <https://www.un.org/es/observances/sign-languages-day>
- Wong, A. (2017). Coding up a neural network classifier from scratch. <https://towardsdatascience.com/coding-up-a-neural-network-classifier-from-scratch977d235d8a24>.
- Workana. (2022). Workana. Obtenido de <https://i.workana.com/glosario/que-es-c/>