UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CARRERA DE INGENIERÍA EN MECATRÓNICA

TRABAJO DE GRADO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN MECATRÓNICA

TEMA:

"***Sistema de detección de uso de cascos de seguridad.***"

AUTOR:

ESCOBAR QUELAL CRISTIAN RENÉ

DIRECTOR:

PhD. DAVID ALBERTO OJEDA PEÑA

Ibarra, 2023

UNIVERSIDAD TÉCNICA DEL NORTE

BIBLIOTECA UNIVERSITARIA

AUTORIZACIÓN DE USO Y PUBLICACIÓN

A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

IDENTIFICACIÓN DE LA OBRA

En cumplimiento del Art. 144 de la Ley de Educación Superior, hago la entrega del presente trabajo a la Universidad Técnica del Norte para que sea publicado en el Repositorio Digital Institucional, para lo cual pongo a disposición la siguiente información:

DATOS DE CONTACTO			
CÉDULA DE IDENTIDAD:	1004849525		
APELLIDOS Y NOMBRES:	Escobar Quelal Cristian René		
DIRECCIÓN:	Ibarra, Bellavista de San Antonio, Barrio Villanueva		
EMAIL:	crescobarq@utn.edu.ec		
TELÉFONO FIJO:	-	**TELÉFONO MÓVIL:**	0969716315

DATOS DE LA OBRA	
TÍTULO:	"SISTEMA DE DETECCIÓN DE CASCOS DE SEGURIDAD"
AUTOR:	Escobar Quelal Cristian René
FECHA:	05 /09 /2023
SOLO PARA TRABAJOS DE TITULACIÓN	
PROGRAMA:	■ PREGRADO POSGRADO
TÍTULO POR EL QUE OPTA:	Ingeniero en Mecatrónica
DIRECTOR:	Ing. DAVID ALBERTO OJEDA PEÑA

CONSTANCIAS

El autor manifiesta que la obra objeto de la presente autorización es original y se la desarrollo sin violar derechos de autores de terceros, por lo tanto, la obra es original, y que es el titular de los derechos patrimoniales, por lo que asume la responsabilidad sobre el contenido de la misma y saldrá en defensa de la Universidad en caso de reclamación por parte de terceros.

Ibarra, a los 05 días del mes de septiembre de 2023.

Nombre: Escobar Quelal Cristian René

Cédula: 1004849525

CERTIFICACIÓN

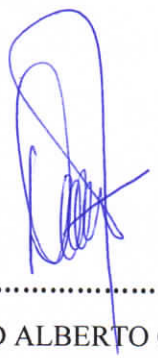
En calidad de tutor del presente Trabajo de Grado titulado: "Sistema de detección de cascos de seguridad", certifico que fue desarrollado por el señor Cristian René Escobar Quelal, bajo mi supervisión.

Ing. David Ojeda

DIRECTOR DEL PROYECTO

APROBACIÓN DEL COMITÉ CALIFICADOR

El Tribunal Examinador del trabajo de titulación "SISTEMA DE DETECCIÓN DE CASCOS DE SEGURIDAD" elaborado por CRISTIAN RENÉ ESCOBAR QUELAL, previo a la obtención del título del INGENIERO EN MECATRÓNICA, aprueba el presente informe de investigación en nombre de la Universidad Técnica del Norte:



.....

PhD. DAVID ALBERTO OJEDA PEÑA

C.C.: 1757898489

.....

PhD. BRIZEIDA NOHEMÍ GÁMEZ APARICO

C.C.: 1758387383

DEDICATORIA

El presente trabajo de titulación principalmente lo dedico a Dios, por haberme ayudado a lo largo de mi vida, dándome la sabiduría y el conocimiento, Él ha sido pilar fundamental para avanzar cada día y permitirme culminar este nivel de estudios de formación profesional. Padre te dedico este trabajo de titulación y sea siempre tu voluntad en mis caminos como profesional.

A mi madre que ha sido la única persona que me ha apoyado incondicionalmente a lo largo de este camino, dándome su amor y cariño, que me ha ayudado a seguir adelante a pesar de que algunos momentos han sido difíciles.

AGRADECIMIENTO

Agradezco a mi madre por su comprensión en los momentos difíciles que se me han presentado en la carrera y a pesar de eso seguir apoyándome. A mi tío Galo por haberme permitido conocer el área técnica, ya que sin él no estuviera en este camino de conocimiento, además de compartir siempre su conocimiento conmigo. A Selena que siempre ha estado conmigo apoyándome.

A mis docentes de la carrera que siempre me han o impartido los conocimientos necesarios que me han ayudado a culminar este gran paso. De manera especial agradezco mi director de tesis, al PhD. David Ojeda por la comprensión a lo largo del desarrollo de este trabajo como tutor.

RESUMEN

En Ecuador se ha presentado una incidencia de accidentes laborales relacionados al mal uso del equipo de protección individual, a pesar de que las industrias proveen de los equipos necesarios para evitar cualquier accidente, existe irresponsabilidad por parte de los trabajadores y usan de manera incorrecta o dejan de usar el casco de seguridad. Viendo esta problemática, en este trabajo de titulación se plantea un sistema de visión artificial basado en redes neuronales convolucionales que permita la detección del uso de cascos de seguridad.

Para su desarrollo se usa la arquitectura YOLO que usa redes neuronales convolucionales para detectar objetos en imágenes de dos dimensiones. Para su entrenamiento se utiliza Google Colab que permite trabajar con GPU's potentes dentro de la nube y se implementa sobre el sistema embebido Jetson Nano 2GB para ejecutar el sistema en tiempo real, el objetivo es que permita la detección de cascos de seguridad dentro de un entorno controlado y emita una señal sonora para que los trabajadores se coloquen correctamente el casco de seguridad.

Con la utilización de este sistema embebido interactuando con un monitor HDMI para la visualización de la detección de cascos de seguridad se busca reducir la incidencia de accidentes laborales que tengan que ver con lesiones en la cabeza. Las pruebas realizadas son en tiempo real al ejecutar el código de detección después de haber entrenado la red YOLO en la Jetson Nano.

ABSTRACT

In Ecuador there has been an incidence of occupational accidents related to the misuse of personal protective equipment, although industries provide the necessary equipment to avoid any accident, there is irresponsibility on the part of workers and they use incorrectly or fail to use the safety helmet. In view of this problem, this degree work proposes an artificial vision system based on convolutional neural networks that allows the detection of the use of safety helmets.

For its development we use the YOLO architecture that uses convolutional neural networks to detect objects in two-dimensional images, for its training we use Google Colab that allows working with powerful GPU's in the cloud and is implemented on the Jetson Nano 2GB embedded system to run the system in real time, the goal is to allow the detection of safety helmets within a controlled environment and emit a sound signal for workers to correctly put on the safety helmet.

The use of this embedded system interacting with an HDMI monitor to visualize the detection of safety helmets is intended to reduce the incidence of occupational accidents involving head injuries. The tests performed are in real time by executing the detection code after training the YOLO network in the Jetson Nano.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

Problema

En el Ecuador las empresas locales han pasado por algunos problemas aplicando las medidas y protocolos de seguridad debido a la irresponsabilidad de algunos operarios dentro de su trabajo o por la falta de políticas de seguridad industrial y salud ocupacional [1], en los registros del La DAEI (Dirección Actuarial, de Investigación y estadística) del IESS, se muestra que el total de accidentes de trabajo es de 15909 en el año 2018, este total está distribuido en porcentajes en las diferentes ramas de actividad económica, las cuales son Servicio Comunal, Social y Personal (24,2%), Industrias Manufactureras (19,3%), Comercio al por Mayor y Menor (15,9%), Agricultura, Silvicultura, Caza y Pesca (13,9%), Establecimientos Financieros, Seguros y Bienes Inmuebles (10,9%), Explotación de Minas y Canteras (1,2%) y finalmente Electricidad, Gas y Agua (2,4%) [2].

Los accidentes son causados por diferentes condiciones riesgosas a las que está expuesto el trabajador; generalmente se producen por protecciones y resguardos inexistentes o no adecuados con el 20,2% de incidencia, Sistemas de advertencia insuficiente con el 11,1%, Equipos de protección individual (EPI) inexistentes o no adecuados y las demás causas son menores al 7%. Por lo antes mencionado han existido diferentes tipos de accidentes laborales e incluso enfermedades profesionales provocadas por el no uso de los equipos de protección, haciendo énfasis en los accidentes que involucran el mal uso de cascos de seguridad dentro de la industria, se han registrado un total de 1537 accidentes de trabajo en las diferentes actividades económicas distribuidos de la siguiente manera:

Servicio Comunal, Social y Personal (364), Industrias Manufactureras (255), Comercio al por Mayor y Menor (257), Agricultura, Silvicultura, Caza y Pesca (212), Establecimientos Financieros, Seguros y Bienes Inmuebles (160), Explotación de Minas y Canteras (30,

Electricidad, Gas y Agua (33), Construcción (67), Transporte, Almacenamiento y Comunicación (78), y finalmente en actividades no definidas (81) [2]

Generalmente las partes más afectadas en los accidentes de trabajo son: el miembro superior (36,0%) y el miembro inferior (26,8%), seguido de ubicación múltiple (12,4%) y cabeza (8,8%); mientras que la parte menos afectada resulta ser el cuello (1,7%) [3]. Tomando en cuenta que la cabeza es una de las partes más afectadas en los accidentes laborales, es importante utilizar siempre un casco de seguridad certificado bajo normas internacionales y nacionales como medio de mitigación en caso de ocurrir un infortunio que involucre caída de objetos, golpes mecánicos, incluso riesgos de naturaleza térmica o eléctrica.

Actualmente en el Ecuador se han comenzado a tomar en cuenta el uso de los equipos de protección personal como una medida necesaria para mantener la seguridad industrial y la salud ocupacional dentro del ambiente laboral [1]. Según [4] la metodología tradicional para detectar el correcto uso de los equipos de protección personal se basaba en la intervención humana de expertos en seguridad ocupacional pero esta metodología tenía varias deficiencias al momento de reunir y analizar la información, por esta razón recientemente se han empezado a usar diferentes tecnologías para detectar el uso de EPP dentro de la industria, entre ellas está la utilización de sistemas de detección basados en visión artificial usando cámaras digitales de bajo costo y plataformas "open source" que facilitan el prototipado rápido de soluciones a este problema que cada año tiene un costo económico, social y ético dentro de la Industria.

La visión artificial es una disciplina engloba todos los elementos y procesos que proporcionen visión a una máquina, [5] se define como la deducción automática de la estructura y propiedades de un mundo tridimensional, posiblemente dinámico a partir de una o varias imágenes tridimensionales. [6] Según Automated Imaging Association, la visión artificial abarca todas las aplicaciones industriales y no industriales en las que una combinación de

hardware y software brinda un guiado operativo a los dispositivos en la ejecución de funciones de acuerdo con la captación y procesamiento de imágenes por medio de un ordenador. Los sistemas de visión artificial requieren de una mayor robustez, fiabilidad para ser implementados de la industria, pero para esto deben cumplir con algunos requisitos como bajo coste, precisión aceptable, resistencia elevada ante entornos hostiles como condiciones mecánicas adversas y temperaturas elevadas, etc.

De este modo se propone la idea de implementar un sistema de detección basado en visión artificial que obtenga las imágenes del entorno y las procese en una computadora para determinar si los operarios están usando o no el casco de seguridad, de esta manera se estará disminuyendo el riesgo de accidentes laborales por objetos en caída libre y además reduciendo costos en atención médica.

Objetivos

Objetivo General:

1. Diseñar un sistema de visión artificial para detectar el uso de cascos de seguridad de los trabajadores.

Objetivos Específicos:

1. Determinar los prerrequisitos y requerimientos para el desarrollo del sistema
2. Diseñar el sistema electrónico y de comunicación
3. Implementar los algoritmos de visión artificial para la detección del uso de los cascos de seguridad
4. Validar el funcionamiento del sistema

Justificación

Esta investigación se basa en el uso y aplicación de herramientas tecnológicas como la visión artificial para evitar accidentes ocasionados por el no uso o uso inadecuado de los cascos de seguridad en el personal de las industrias. Los accidentes laborales y enfermedades profesionales generan un costo equivalente al 3,94% del Producto Interno Bruto (PIB) global de cada año de diversos países, de acuerdo con la Dirección de Seguro General de Riesgos de trabajo del IEES, mostró que en 2018 los gastos generados por incapacidad temporal o permanente son de 36 millones de dólares, pero más allá de los gastos económicos están la afectación y la implicación psicológica para los involucrados en un accidente laboral, siendo estos el empleador, el trabajador y su entorno familiar. Por esta razón se plantea implementar un sistema que permita la detección del incumplimiento de las normas de seguridad, en este caso que detecte el uso de cascos de seguridad, la aplicación de este tipo de sistemas en la industria garantiza disminuir los accidentes y el riesgo laboral e incluso el fallecimiento del personal y como consecuencia reducirá también el gasto en atención médica considerablemente.

Alcance

El presente trabajo de grado propone construir el prototipo de un sistema para la detección del uso de cascos de seguridad aplicable a un entorno laboral del área industrial sujeto a riesgo de caída de objetos. Se propone utilizar algoritmos de visión artificial para la detección de objetos que por medio de una o varias cámaras que enviarán las imágenes al computador para determinar el uso incorrecto y/o el no uso de los cascos de seguridad. El sistema será un prototipo conformado por una cámara web capaz de reconocer si un trabajador está usando un casco de seguridad y además emite una señal sonora para recordarle al operario de colocárselo. La aplicación o software del sistema será desarrollada en PC entrenando una red neuronal y trabajará sobre un microcontrolador que sea capaz de procesar imágenes.

CAPÍTULO I

1. MARCO TEÓRICO REFERENCIAL

Considerando la problemática planteada, el objetivo general de este trabajo de grado es desarrollar un sistema de visión artificial que permita identificar el correcto uso de cascos de seguridad por parte de los trabajadores. Con este enfoque, se busca enfrentar problemas ocasionados en lugares de trabajo, especialmente en lo que respecta a los accidentes laborales y el cumplimiento de las medidas de protección personal en la industria. A través de una implementación de tecnología, como la visión artificial, se pretende reducir la cantidad de accidentes laborales o prevenirlos al asegurar el adecuado uso de los cascos de seguridad por parte de los trabajadores.

A continuación, se presentarán los trabajos relacionados con anterioridad referentes al tema de estudio y algunos conceptos clave que resultan fundamentales para entender el tema de investigación que se abordará en este trabajo.

1.1.Antecedentes

M. Massiris en su trabajo de investigación "DETECCIÓN DE EQUIPOS DE PROTECCIÓN PERSONAL MEDIANTE RED NEURONAL CONVOLUCIONAL YOLO" diseñó un detector de imágenes que permite reconocer el uso de cascos, guantes y ropa de seguridad basado en Deep Learning y redes neuronales convolucionales y llegó a buenos resultados haciendo que el sistema sea robusto en varios escenarios y condiciones, además es relativamente rápido, aunque aún deben refinar más el proceso de detección para eliminar falsos positivos y así aumentar su tasa de aciertos [4].

X. Wang en su artículo "A Safety Helmet and Protective Clothing Detection Method based on Improved-YoloV3" trabaja en un método que permita la detección de ropa de protección

personal y de cascos de seguridad en la industria de la construcción, los datos con los que trabaja son videos e imágenes obtenidas de internet pero al mismo tiempo obtiene algunas imágenes de construcciones locales, para el filtrado y entrenamiento de la red neuronal y obtiene los resultados esperados después de categorizar las imágenes haciendo este método confiable y rápido para la detección de los cascos y la ropa de seguridad [7].

F. Wilhelm en su investigación "Detecting motorcycle helmet use with deep learning" implementa un algoritmo de detección que permite determinar si uno o varios ocupantes de una motocicleta llevan el casco de seguridad, para ello usa la red convolucional YOLO9000 y la entrena usando videoclips e imágenes grabadas con una cámara de resolución 1920x1080 a 10 cuadros por segundo y usa ImageNet para determinar el número de ocupantes y si llevan o no cascos de motociclista, como resultado y al ser un módelo basado en imágenes dinámicas obtiene una efectividad del 72.3% pero es eficiente y permite que el sistema detecte el no uso de cascos, esto ayuda a reducir los accidentes de trásito que ocasionan muchas muertes alrededor del mundo [8] .

Usando visión artificial para la detección del uso de equipos de protección personal en las empresas se reduce considerablemente el impacto de los accidentes laborales, pero para lograr eso es importante garantizar la confiabilidad del sistema antes de implementarlo dentro de la industria.

1.2.Marco Teórico

1.2.1 Actividades Económicas en el Ecuador

El ecuador tiene una económica diversificada con tres sectores importantes. Según los datos del Instituto Nacional de Estadísticas y Censos (INEC) de Ecuador del año 2020, el porcentaje de personas de la Población Económicamente Activa (PEA) que trabajan en el sector primario, secundario y terciario es el siguiente [9]:

1.2.1.1 Sector primario

Alrededor del 25,2% de la PEA trabaja en este sector, que incluye actividades como agricultura, pesa, ganadería, minería y explotación forestal [9].

1.2.1.2 Sector secundario

Alrededor del 25,4% de la PEA trabaja en este sector, que incluye actividades como la industria manufacturera, la construcción y la producción de energía [9].

1.2.1.3 Sector terciario

Alrededor del 49,4% de la PEA trabaja en este sector, que incluye actividades como los servicios, el comercio, el turismo, la educación y la salud [9].

Las Figuras 1 y 2 muestran el porcentaje de participación de la PEA en las diferentes actividades de los principales sectores económicos del ecuador.

Figura 1

Participación por sector en relación con la Población Económicamente Activa (PEA), [2].

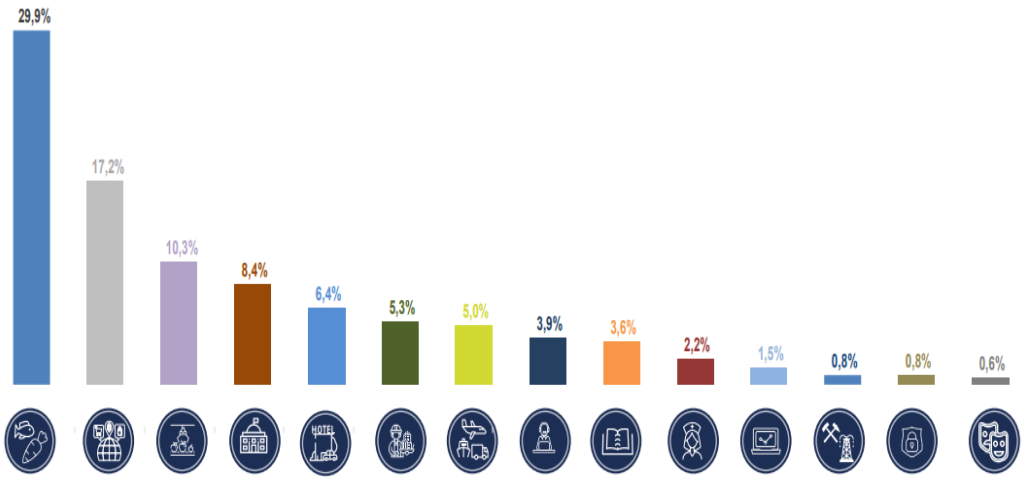


Figura 2

Descripción de la simbología empleada anteriormente en relación de la PEA, [10].

En base a lo anterior, cada sector productivo presenta un ambiente laboral diferente en la PEA estará expuesta a riesgos laborales en donde requerirán el uso de Equipo de Protección Personal (EPP) en función de los riesgos a los que estén expuestos para preservar la salud o integridad física de los trabajadores [11].

1.2.1.4 Ambiente laboral

El ambiente laboral puede ser un factor determinante en la aparición de peligros laborales en el lugar de trabajo. El ambiente laboral puede referirse a las condiciones físicas, químicas, biológicas y psicosociales que rodean el lugar de trabajo y que pueden tener un impacto significativo en la salud y seguridad de los trabajadores [12].

Entre los factores ambientales que pueden contribuir a los peligros laborales se encuentran:

- **Iluminación:** Una iluminación inadecuada en el área de trabajo aumenta el riesgo de accidentes laborales, como tropezones, resbalones y caídas [12].
- **Ruido:** Niveles de ruido excesivos puede causar problemas de audición, fatiga y estrés, y también puede interferir con la capacidad de los trabajadores para comunicarse y concentrarse [12].

- **Temperatura:** La exposición a temperaturas extremas puede provocar enfermedades relacionadas con el calor y el frio, así como también aumentar el riesgo de accidentes laborales [12].
- **Carga de trabajo:** la carga de trabajo excesiva y la presión por cumplir plazos pueden aumentar el estrés y la fatiga, lo que puede aumentar el riesgo de accidentes laborales [12].

1.2.2 Enfermedades y accidentes laborales

1.2.2.1 Enfermedades laborales

Según el Reglamento General de la Ley Orgánica de Seguridad y Salud en el Trabajo de Ecuador, se considera enfermedad laboral a toda patología derivada de la exposición a factores de riesgo presentes en el ambiente de trabajo o ejercicio de profesión. Además, se establecen los siguientes requisitos para que una enfermedad sea considerada laboral [13].

Para que una enfermedad sea considerada de carácter laboral se debe cumplir dos requisitos fundamentales [14]:

- Que exista una relación causa efecto con la relación laboral desarrollada.
- Que la enfermedad haya sido contraída como consecuencia de la exposición a factores de riesgo presentes en el ambiente de trabajo o en el ejercicio de la profesión.
- Que la enfermedad sea reconocida como enfermedad laboral por la autoridad competente.

Además, la normativa establece que el empleador tiene la responsabilidad de implementar medidas de prevención y control de los factores de riesgo presentes en el ambiente de trabajo, con el fin de reducir la probabilidad de padecer enfermedades laborales [14].

1.2.2.2 Accidentes en el trabajo

Se entenderá por accidente laboral toda lesión corporal que sufre un trabajador por cuenta ajena con ocasión o como consecuencia de su trabajo [15].

Tendrán la consideración de accidentes de trabajo:

- Los que sufra el trabajador al ir o al volver del lugar de trabajo.
- Los que sufra el trabajador con ocasión o como consecuencia de las tareas que, aun siendo distintas a las de su categoría profesional ejecute el trabajador en cumplimiento de las órdenes del empresario o espontáneamente en interés del buen funcionamiento de la empresa.
- Las enfermedades no incluidas en el cuadro de enfermedades profesionales que contraiga el trabajador con motivo de la realización de su trabajo, siempre que se pruebe que la enfermedad tuvo por causa exclusiva la ejecución de este.

1.2.2.3 Seguridad y salud en el trabajo

La gestión de la seguridad y la salud forma parte de la gestión de una empresa. Las empresas deben hacer una evaluación de los riesgos para conocer cuáles son los peligros y los riesgos en sus lugares de trabajo, y adoptar medidas para controlarlos con eficacia, asegurando que dichos peligros y riesgos no causen daños a los trabajadores [16].

1.2.3 Riesgos Laborales

Se entiende por riesgos laboral el conjunto de factores físicos, psíquicos, químicos, ambientales, sociales y culturales que actúan sobre el individuo; la interrelación y los efectos que producen esos factores dan lugar a la enfermedad ocupacional [17].

1.2.3.1 Clasificación

- **Riesgos del ambiente:** iluminación, ventilación, ruido, humedad, temperatura.
- **Riesgos contaminantes:** producidos por sustancias físicas, químicas o biológicas.
- **Factores de inseguridad:** caídas de objetos debido a deficiencias en la construcción, ausencia o mecanismos de seguridad.

- **Sobrecarga muscular:** producido por ejercer grandes esfuerzos y está sometido a sobrecarga física o una situación de trabajo inadecuada.

1.2.3.2 Tipos de riesgo laborales

- **Físico:** son aquellos riegos en el que los factores ambientales que dependen de las propiedades físicas del cuerpo actúan sobre los tejidos y órganos del cuerdo, tales como carga física, ruido, iluminación, radiación, entre otros.
- **Mecánico:** son el conjunto de factores que puedan dar lugar a una lesión, producidos por maquinaria, herramientas, aparatos de izar, instalaciones, superficies de trabajo, etc.
- **Químico:** son aquellos factores derivados de una exposición de agentes químicos.
- **Biológico:** la posible exposición a microorganismos que puedan dar lugar a enfermedades, motivada por la actividad laboral.
- **Ergonómico:** aquellos que se originan cuando el trabajador interactúa con su puesto de trabajo y cuando las actividades laborales presentan movimientos, posturas o acciones que pueden producir daños a su salud.
- **Psicosocial:** se producen cuando hay monotonía del trabajo, sobrecarga laboral, minuciosidad de la tarea, alta responsabilidad, autonomía en la toma de decisiones, supervisión y estilos de dirección deficiente.

1.3.Equipo de Protección Personal

Un EPP es un equipo que protege al usuario del riesgo de accidentes o de efectos adversos para la salud. Puede incluir elementos como cascos de seguridad, guantes, protección de los ojos, prendas de alta visibilidad, calzado de seguridad, arneses de seguridad y equipos de protección respiratoria [18].

El artículo 51 del Reglamente de Seguridad y Salud de los Trabajadores y Mejoramiento de Medio Ambiente establece lo siguiente:

"Uso de los elementos de protección personal. El empleador debe proporcionar y exigir a los trabajadores la utilización de los elementos de protección personal adecuado para cada actividad, con el fin de proteger su integridad física. Estos elementos no deben ser objeto de costos para el trabajador".

Existen varias razones por las cuales algunos trabajadores pueden no usar correctamente o no usar en absoluto EPP en su lugar de trabajo [18].

- **Falta de conciencia:** algunos trabajadores no son conscientes de los riesgos asociados con su trabajo y la importancia de usar EPP para su protección.
- **Incomodos o restrictivos:** algunos pueden encontrar que los EPP son incomodos o restrictivos, lo que dificulta su uso durante largas horas de trabajo.
- **Falta de entrenamiento:** puede llevar a que a los trabajadores a no utilizarlos correctamente o a no utilizarlo en absoluto.
- **Falta de supervisión:** en algunos casos, los supervisores pueden no hacer cumplir el uso de los EPP, lo que lleva a los trabajadores a no usarlos.
- **Percepción de riesgo bajo:** los trabajadores pueden subestimar el riesgo asociado con su trabajo y pueden no considerar necesario usar EPP.

Los equipos de protección personal para protección frente a riesgos físicos varían dependiendo del tipo de riesgo físicos varían dependiendo del tipo de riesgos al que están expuestos los trabajadores. La Figura 3 muestra los EPP más comunes.

Figura 3

Equipo de Protección Personal más comunes [19].

En los trabajos con riesgos de exposición mecánicos, eléctricos, químicos o térmicos, los trabajadores deberán emplear equipos de protección para pies y piernas, como calzado aislante de la electricidad, suelas aislantes del frío o calor, protector del metatarso, plantillas resistentes a la perforación, rodilleras o empeines resistentes al corte, entre otros.

En los trabajos con exposición a riesgos eléctricos se requieren de diversos equipos aislantes de la electricidad para todo el cuerpo, como cascos aislantes para la cabeza, pantallas faciales, guantes, calzado aislante eléctrico, etc.

1.2.4 Protección de la cabeza: casco de seguridad

El casco de seguridad es un equipo de protección personal que es utilizado en múltiples industrias donde los trabajadores están expuestos a riesgos de lesiones en la cabeza debido a la caída de objetos, golpes, cortes y otros accidentes.

La importancia del casco de seguridad radica en que protege la cabeza y el cerebro de lesiones graves o fatales que pueden resultar de un impacto o una caída. Además, es un requisito legal en muchas industrias para proteger a los trabajadores de posibles accidentes [20].

En la Figura 4 se indica los colores y significado más comunes en los cascos de seguridad, con la cual se puede identificar visualmente a los trabajadores y su rol en la obra o proyecto [21].

Figura 4

Cascos de seguridad según la actividad a desempeñar [22].

En la siguiente tabla se muestra la asignación de colores para las diferentes asignaciones en el campo laboral de los trabajadores:

Tabla 1

Codificación de los cascos de seguridad **[23]**

Codificación de colores de los cascos de seguridad	
Blanco	Arquitectos, ingenieros, jefes de obras y rangos altos.
Gris	Estudiantes o visitantes de obra.
Amarillo	Personal operativo.
Verde	Profesionales de higiene y seguridad o de servicio médico.
Rojo	Brigadas de emergencia, inspectores de seguridad y bomberos.
Azul	Personal de electricidad.

1.2.4.1 Control del uso de EPP

El control del uso de EPP se realiza mediante diversas medidas, entre las cuales se encuentran [24]:

- Capacitación: los empleados deben ser capacitados sobre la importancia del uso de EPP y cómo utilizarlo correctamente.
- Supervisión: los supervisores deben realizar inspecciones regulares en el lugar de trabajo para asegurarse de que los trabajadores estén utilizando los EPP adecuados.
- Sanciones: en el caso de que un trabajador no utilice el EPP adecuado, pueden aplicarse sanciones disciplinarias.

Existen diferentes razones por las cuales puede fallar la supervisión en el control de EPP [24]:

- Falta de capacitación: si los supervisores no han recibido capacitación adecuada sobre el uso y la importancia del EPP, pueden no saber cómo identificar si los trabajadores lo están usando correctamente o no.

- Falta de seguimiento: si los supervisores no realizan un seguimiento constante del uso del EPP por parte de los trabajadores, puede haber momentos en los que los trabajadores no lo usen y no sean detectados.
- Falta de liderazgo: si los lideres de la empresa no muestran compromiso claro con la seguridad y la importancia de uso del EPP, esto puede transmitirse a los supervisores y trabajadores.

1.2.5 Sistemas de detección

Un sistema de detección es un conjunto de componentes que operan con señales de entrada para ejecutar una determinada función, dando lugar a una salida que refleja un resultado esperado. En otras palabras, es un sistema que trabaja recibiendo una señal de entrada y produce una respuesta que muestre un resultado esperado. A continuación, se detallan algunos sistemas que pueden ser usados para detectar EPI'S.

1.2.5.1 Sistemas basados en sensores

Según [25] la detección de EPI puede aplicarse usando sistemas basado en sensores que incorporan la tecnología RFID para mejorar la seguridad de los trabajadores, este sistema consiste en realizar un etiquetado de los equipos de protección personal, como botas, guantes, cascos, mascarillas, etc. y monitorear mediante antenas su uso para verificar que llevan puestos los EPI'S obligatorios, con esto se controla el acceso a los recintos de trabajo y también el bloqueo de maquinaria en caso de no usar o hacer un uso incorrecto de los EPI'S.

En el área industrial existen zonas peligrosas donde es difícil acceder y además se debe hacer con extremada precaución, de esta manera, se controla el acceso a las zonas de riesgo y, en caso de que un trabajador no utilice o utilice incorrectamente los EPI's, si se detecta que un trabajador ingresa sin las normas adecuadas de seguridad se activa una alarma para avisar al personal de supervisión o bloquea directamente la maquinaria [25].

1.2.5.2 Sistemas integrados con cámaras inteligentes

Los sistemas integrados con cámaras inteligentes son una tecnología que usa la capacidad de cámaras que integran algoritmos y software avanzados para el análisis de daros en tiempo real. Estos sistemas se utilizan en una variedad de aplicaciones en diversos campos, como seguridad, industria, transporte, salud, entre otros. En el ámbito industrial, en los últimos años se ha implementado este tipo de sistemas con el objetivo de mejorar los procesos y mejorar la producción dentro de la Industria 4.0 ya que son capaces de corregir todo tipo de errores en una línea de producción. Estos sistemas cuentan con una tecnología que les permite tomar decisiones que hacen que el sistema funcione de forma autónoma y bajo parámetros de trabajo específico configurados por el usuario [26].

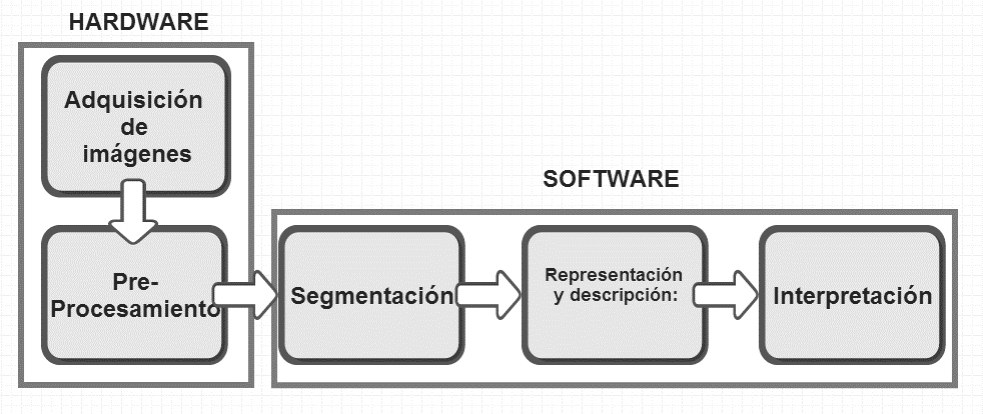
1.2.5.3 Sistemas de visión por computador avanzados

Un sistema de visión por computador avanzado se caracteriza por su potencia en el hardware y el software que utiliza, siendo este tipo de sistema perfecto para analizar grandes cantidades de datos al mismo tiempo. En consecuencia, un sistema de visión avanzada puede ser capaz de realizar operaciones de detección con un nivel de complejidad alto e inclusive suelen tener una capacidad cognitiva que le permite interpretar datos, aprender y emplear esos conocimientos para realizar tareas acordes a lo que fueron programados y conseguir metas concretas dependiendo de su nivel de adaptación [26].

Los sistemas de visión avanzados actualmente se componen de diferentes bloques ordenados para formar una estructura que permite capturar las imágenes de entrada para su posterior procesamiento, su funcionamiento se puede observar en el siguiente diagrama:

Figura 5

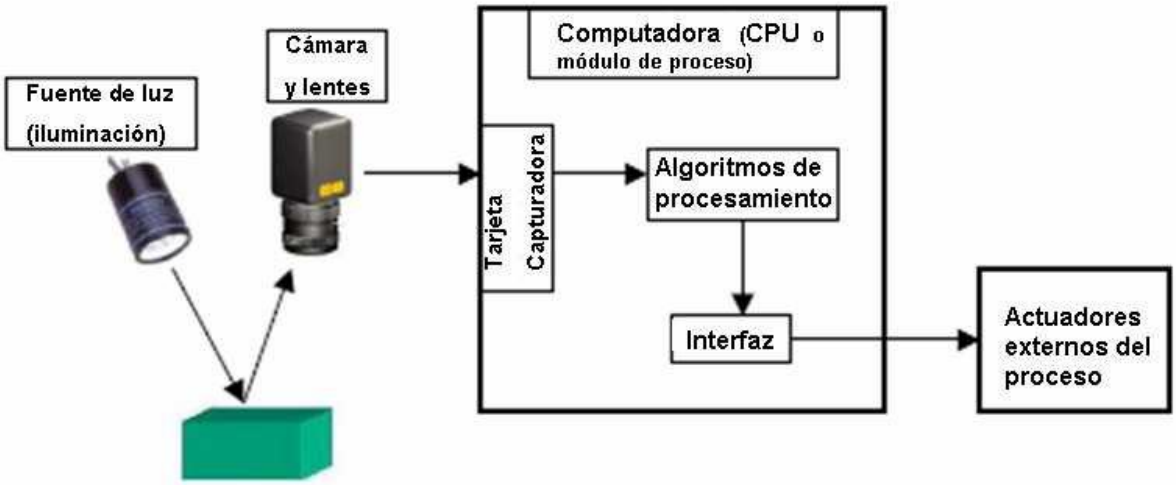
Estructura de un sistema de vision artificial actual [27].



En la figura 5 se puede ver una guía de los pasos que debe tomar un módulo de visión para poder lograr su función.

Figura 6

Sistemas de visión avanzados en tiempo real **[28]**.



En Figura 6, se observa cómo se ejecuta básicamente un sistema de visión artificial en tiempo real implementado en un sistema de manufactura para la industria 4.0.

1.2.6 Visión artificial

La visión artificial, también conocida como visión por computadora o visión de máquina, es un campo de la inteligencia artificial (IA) que se enfoca en permitir que las computadoras interpreten y comprendan la información visual del mundo que las rodea, de manera similar a la visión humana. Esto implica el uso de algoritmos y modelos de aprendizaje profundo para analizar y extraer significado de imágenes o datos de video.

La visión artificial tiene muchas aplicaciones en diversas industrias, incluyendo la salud, automotriz, robótica y seguridad [29]. La visión por computador provee soluciones reales y eficientes; que se logra al procesar, analizar e interpretar un conjunto secuencial de datos en una entrada ya sean estas de imágenes o de video.

La visión artificial dentro de la inteligencia artificial se basa en un conjunto de modelos y técnicas, las cuales permiten realizar un procesamiento, análisis y la caracterización de cualquier tipo de información que se obtenga a través de imágenes digitales. En el procesamiento de las imágenes, se realiza un análisis secuencial de las características, con la finalidad de incrementar su calidad y extraer información que permita una comprensión automática de algún patrón o característica.

1.2.7 Procesamiento de Imágenes

El procesamiento digital de imágenes, también llamado manejo de imágenes por computador es una disciplina que se encarga de procesar y analizar imágenes del mundo real con la finalidad de ser interpretadas por un ordenador. El procesamiento de imágenes puede considerarse una rama de la visión por computador, la cual agrupa varias ciencias como la óptica, electrónica, matemáticas, fotografía e informática [30].

Según [30] cada una de estas ciencias aporta varios factores que se combinan para indicar la tendencia futura que tendrá el procesamiento de imágenes. El principal factor que influye en

cómo se desarrolla el campo de la visión por computador es el costo de poder de cómputo que cada vez más disminuye con los ordenadores actuales en comparativa con las primeras computadoras.

Un segundo factor crucial en el crecimiento del procesamiento digital de imágenes es el aumento de equipos de digitalización de imágenes. Además, existen nuevas tecnologías que se han ido incorporando a lo largo de los años a este campo, entre estas se tiene las CCD (Dispositivo de Carga Acoplada, por sus siglas en inglés) de alta resolución, redes neuronales y procesadores matriciales [30].

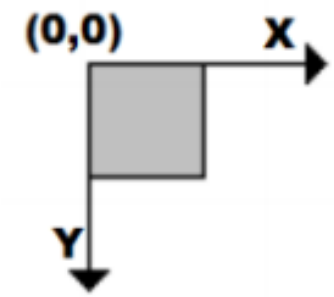
Teniendo claro a que se refiere el procesamiento de imágenes, se puede hacer una diferenciación entre procesamiento digital y la visión artificial. La visión artificial es una disciplina que trata de emular la vista humana con el uso de computadoras, usando el aprendizaje automático y pudiendo realizar inferencias en tiempo real basándose en entradas visuales. Con base en esta distinción, se puede afirmar que el procesamiento digital de imágenes es parte de la visión por computador, y a su vez, la visión por computador parte de un área más amplia denominada Inteligencia Artificial. [31]

1.2.7.1 Representación y definición de una Imagen

Una imagen digital puede ser definida como una función de dos dimensiones, $f(x, y)$, donde (x, y) son las coordenadas cartesianas en el espacio y el valor de f en el punto (x, y) representa la intensidad o valor de nivel de gris de la imagen en ese punto. Una imagen digital además puede considerarse como una matriz de filas y columnas, cuyos índices indican el punto de la imagen y el valor del elemento correspondiente. A este tipo de distribución digital se le conoce como elementos de la imagen o comúnmente se le conoce con el nombre de pixeles [32].

Figura 7

Descripción de una imagen digital en 2D [32]



Las imágenes digitales también tienen características, también llamadas propiedades que están relacionadas íntimamente, estas son: la resolución espacial, la resolución de niveles y el número de pasos [32].

A continuación, se describen las propiedades de una imagen digital:

- Resolución espacial: se refiere al número de pixeles que tiene por unidad de longitud, significa que una imagen de m filas y n columnas va a tener $n \ x \ m$ pixeles, se relaciona directamente con los detalles que puedan verse dentro de la imagen [32].
- Resolución por niveles: también llamada profundidad de píxel, indica el número de bits que existen dentro de un píxel, sirve para indicar el número de niveles de gris que pueden apreciarse en la imagen [32].
- Numero de planos: es el número de arreglos de pixeles que componen la imagen. Por ejemplo, una imagen de tonos de grises está compuesta por un solo plano, mientras que el color verdadero está compuesta por tres planos de color, el rojo, verde y azul [32].

1.2.8 Redes Neuronales

Según [33] las redes neuronales artificiales RNA están inspiradas en la biología, esto significa que están formadas por elementos que se comportan de manera análoga a las neuronas (en las funciones más elementales) y están organizadas de una forma similar a la del cerebro, pero las analogías no son muchas más.

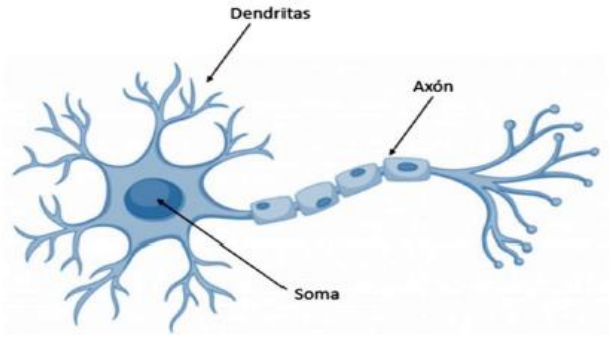
Las características fundamentales de las RNA son:

- Aprenden de la experiencia: pueden modificar su comportamiento dependiendo de la respuesta de su entorno.
- Generalizan de ejemplos anteriores a los ejemplos nuevos: cuando ha sido entrenada una RNA la respuesta de la red puede ser hasta cierto punto sensible a pequeñas variaciones en la entrada, haciendo que sean idóneas para el reconocimiento de patrones.
- Abstracción de la esencia de las entradas: algunas RNA son capaces de abstraer información, significa que una vez la red fue entrenada puede reconocer información en patrones distorsionados, es decir, será capaz de dar un resultado correcto, aunque ante una entrada que la red nunca había visto [33].

La analogía que existe entre una neurona biológica y una RNA radica en su estructura que comparte características similares. En la figura 8 y 9 se puede ver esta analogía:

Figura 8

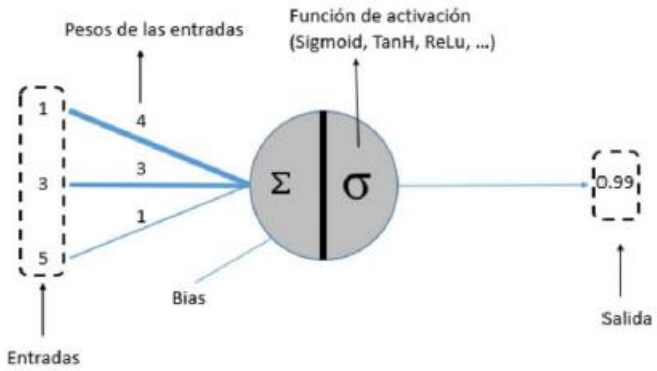
Neurona biológica con sus partes [34]



En la Figura 8, se observa cómo funciona la neurona, la cual está compuesta por: dendritas, que captan los impulsos nerviosos; el soma procesa estos impulsos y el axón emite el impuso de respuesta a las neuronas contiguas. [34]

Figura 9

Estrucura básica de una red neuronal artificial **[34]**



En la Figura 9, se representa el esquema análogo de una neurona artificial, donde la suma de las entradas se multiplica por sus pesos asociados, determina el "impulso nervioso" que se recibe. A este valor que recibe la célula se le conoce como función de activación y al valor que devuelve se le llama salida de la neurona [32].

El cerebro humano este compuesto de neuronas conectadas entre sí, al igual que una red neuronal está formada por capas conectadas. Una capa es un conjunto de neuronas donde sus

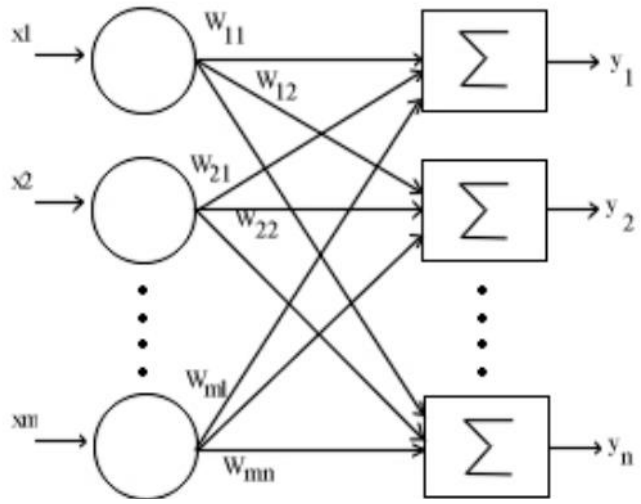
entradas provienen de una capa exterior de datos, en el caso de ser la primera capa se llaman datos de entrada y sus salidas se reflejan en la siguiente capa [34].

Las redes neuronales se pueden clasificar en dos tipos principales, las redes de capa simple y las redes multicapa:

- **Red de capa simple:** este tipo de red se compone de una sola neurona capaz de realizar modelos simples de funciones, para incrementar su productividad se organiza en redes de estas y puede formar un patrón de entradas que proporciona una salida. La representación gráfica de la red multicapa se muestra en la figura n:

Figura 10

Red Neuronal de capa simple **[33]**



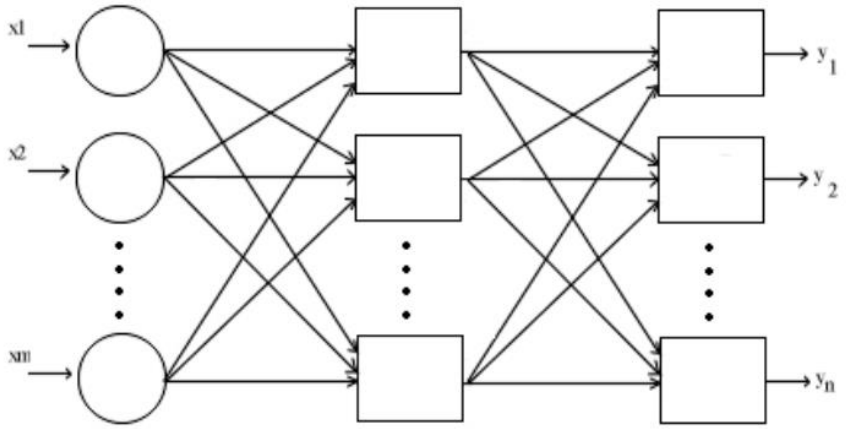
En la figura 10 se puede observar el modelo de una red de capa simple donde se añade una capa inicial que no es contabilizada a efectos de computación puesto que esta sirve solo para distribuir las entradas de los perceptrones, por esta razón se le denomina capa 0 [33].

- **Red multicapa:** Una red multicapa se forma por el conjunto de redes de capa en cascada unida por los pesos, donde la salida de una capa es la entrada de la siguiente

capa. Este tipo de red es capaz de aprender funciones que una red de una capa no es capaz, la única diferencia es que para esta red multicapa el poder computacional que se necesita aumenta ya que suele tener una función no lineal entre las capas, porque generalmente se usa una función de activación sigmoidea [33]. La representación gráfica de este tipo de red se observa en la figura 11:

Figura 11

Estructura de una red neuronal multicapa **[33]**



1.2.9 Redes neuronales convolucionales

Las redes neuronales convolucionales CNN son una arquitectura de red neuronal artificial más avanzada que se deriva como variación del perceptrón multicapa y se diferencia de una RNA convencional porque usa capas de convolución y de subsampling para procesar de mejor manera imágenes y videos de entrada [35].

Este tipo de arquitectura de red está diseñada especialmente para trabajar con datos de dos dimensiones, por esta razón se utilizan mucho en el reconocimiento de imágenes y señales de voz en espectrogramas. Además, si se modifica pueden trabajar en más de dos dimensiones. Fue desarrollada en 1982 por Kunihiko Fukushima cuando se encontraba trabajando en una red neuronal que imita el proceso de córtex visual. En 1998 Yann LeCun

entreno una red llamada LeNet que empleaba una estructura de red neuronal convolucional consiguiendo clasificar dígitos con una precisión del 99.3%. Pero no fue hasta el 2012 cuando fue presentada la red AlexNet de Geoffrey Hinton donde se observó realmente la capacidad de potencia que tenían este tipo de redes para la clasificación de imágenes [35].

12.1 Estructura y propiedades de las CNN

La arquitectura de una red neuronal convolucional comienza con la aplicación de capas convolucionales y de pooling, cuando todos los elementos se agrupan como un vector (proces o de aleteo), repite este proceso hasta obtener un conjunto de matrices. Muchos elementos co ntienen un conjunto de elementos computacionalmente viables que se introducen como entra das a una red neuronal [35].

12.2 Componentes de la arquitectura de una CNN

12.1 Capa convolucional

Se refiere a una capa principal dentro de una red neuronal, siendo indispensable en una CNN tener una o más de estas. Esta capa es donde se colocan los pesos de la red que se va a entrenar. [36]

12.2 Capa de agrupación (Pooling Layer)

La capa de pooling (también conocida como capa de submuestreo o reducción) es un componente clave en las redes neuronales convolucionales (CNN) utilizadas para el procesamiento de imágenes. Su función principal es reducir la dimensionalidad espacial de las características mientras conserva la información más relevante.

1.2.3 Capa de unidades lineales rectificadas ReLu

La capa de unidades lineales rectificadas (ReLU, por sus siglas en inglés Rectified Linear Units) es una función de activación ampliamente utilizada en redes neuronales, incluyendo las redes convolucionales (CNN).

1.2.10 YOLO

YOLO (You Only Look Once) es un popular algoritmo de código abierto creado para la detección de objetos en tiempo real y desarrollado por Joseph Redmon, Santosh Divvala, Ross Girshick y Ali Farhadi. La principal característica distintiva de YOLO es que realiza la detección de objetos en una sola pasada (una sola mirada) a través de la red neuronal, a diferencia de otros métodos que requieren múltiples regiones de interés y pasos para detectar objetos [37].

El enfoque de YOLO para la detección de objetos se basa en convertir el problema en un problema de regresión, en el cual la red predice directamente las coordenadas de los cuadros delimitadores (bounding boxes) y las probabilidades de las clases asociadas en una sola salida. Es decir, la red toma una imagen de entrada, la procesa a través de capas convolucionales, y produce una salida que contiene información sobre los cuadros delimitadores y las clases de los objetos detectados [37].

Las características clave de YOLO incluyen [38]:

1. **Eficiencia:** Debido a que YOLO realiza la detección en una sola pasada, es muy rápido y eficiente, lo que lo hace adecuado para aplicaciones en tiempo real, como detección de objetos en videos o en sistemas de vigilancia.

2. **End-to-End:** YOLO es un modelo end-to-end, lo que significa que no requiere un postprocesamiento complejo ni la combinación de múltiples etapas para obtener las predicciones de detección.

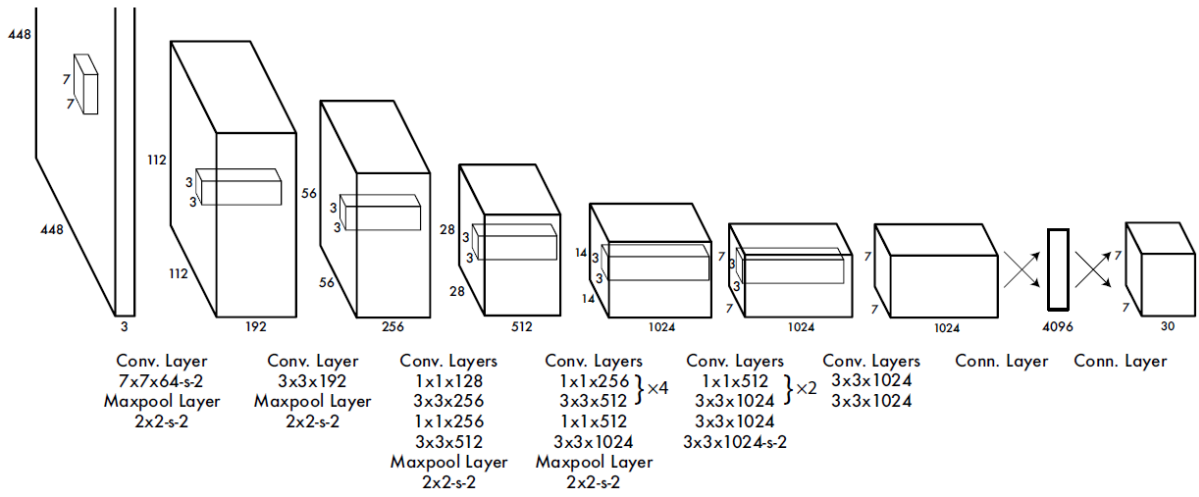
3. **Detección múltiple:** A diferencia de otros enfoques de detección de objetos que pueden tener problemas para detectar objetos superpuestos o cercanos, YOLO puede detectar múltiples objetos en una sola imagen, incluso si se superponen.

4. **Generalización:** YOLO generaliza bien a diferentes tipos de objetos y escenarios, lo que lo hace adecuado para una amplia gama de aplicaciones.

A lo largo del tiempo, han surgido diferentes versiones de YOLO, cada una con mejoras y modificaciones en la arquitectura original que se muestra en la Figura 12. Por ejemplo, YOLOv2 y YOLOv3 incorporan mejoras como el uso de detección de múltiples escalas y la utilización de capas de convolución más profundas.

Figura 12

Arquitectura de YOLO CNN [38]



YOLO ha sido ampliamente adoptado y ha demostrado un buen rendimiento en tareas de detección de objetos, lo que la convierte en una herramienta popular para aplicaciones de visión por computadora.

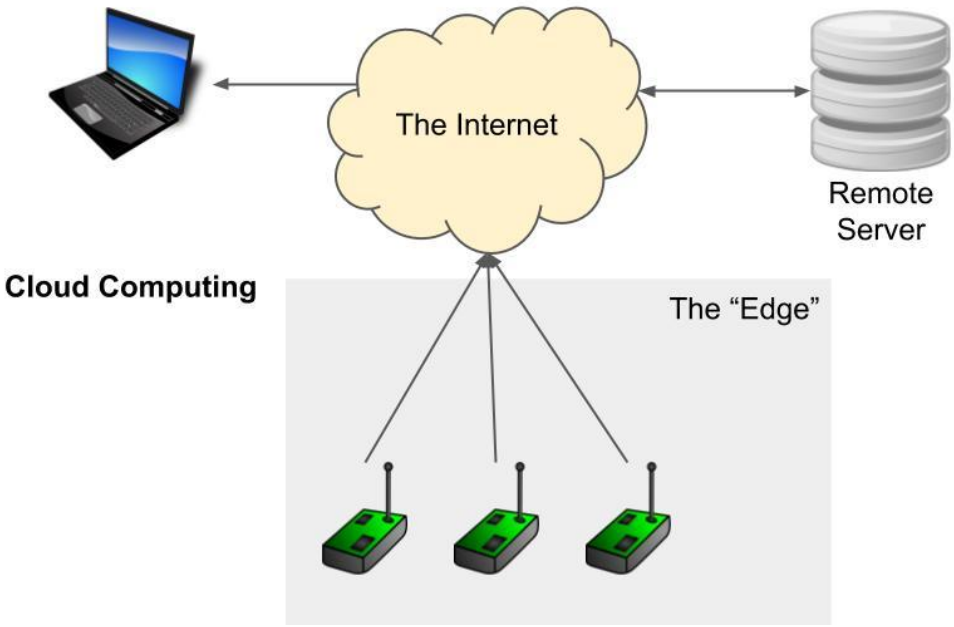
1.2.11 Edge AI

El concepto Edge IA se refiere a la utilización de la IA en el borde, es decir que los cálculos que realice una IA sean cerca de los usuarios, en el extremo de la red, en lugar de en ubicación

centralizada como el centro de datos de un proveedor de servicios en la nube. Con el aumento de dispositivos de computación perimetral, el potencial de la inteligencia artificial ha ido aumentando, siendo capaz de ejecutar sistemas que funcionen con inteligencia artificial en pequeños kits de desarrollo que tienen un poder computacional decente [39].

Figura 13

Estructura general del Edge AI [40]



El enfoque que tiene Edge AI como se observa en la figura 12, se centra en la ejecución algoritmos y modelos de inteligencia artificial en dispositivos periféricos que se encuentren junto al usuario justo al borde de la red. Estos dispositivos por lo general estos dispositivos suelen ser de recursos limitados, pero al estar situados al borde reducen la latencia asociada con sistemas que realizan cálculos en la nube. Además, se reconoce el potencial que tiene el uso de periféricos en entornos locales para la ejecución de algoritmos de aprendizaje automático e inferencia. [39].

14.1 Ventajas del Edge AI

La idea central del Edge AI es llevar a cabo el procesamiento y análisis de datos más cerca de la fuente de donde se generan estos datos, teniendo las siguientes ventajas:

- Menor Latencia: Al procesar datos directamente en un dispositivo local la latencia disminuye, ya que no se envían los datos a un servidor y esperar su procesamiento [41].
- Mayor privacidad y seguridad: Los datos se mantienen dentro del dispositivo en el que se ejecuten los algoritmos de AI, se reduce la necesidad de compartir datos en la nube, esto mejora significativamente la privacidad y seguridad de la información [41].
- Menor carga en la red: El procesamiento se realiza localmente, no se necesario usar el tráfico de red, esto es beneficioso en el caso de tener un ancho de banda limitado o conexiones inestables [41].
- Funcionamiento offline: Esto permite ejecutar algoritmos de AI localmente, procesar datos y obtener inferencias de modelos que se haya entrenado con anterioridad [41].

1.2.11.1 Aplicaciones del Edge AI

- **Sistemas de seguridad:** Poder detectar intrusos o incidentes de manera autónoma y enviar alertas en tiempo real [41].
- **Robots industriales:** los robots con Edge AI pueden realizar tareas de manera autónoma y adaptarse a cambios en el entorno sin intervención humana [41].

- **Dispositivos médicos:** los dispositivos médicos equipados con Edge AI pueden analizar y procesar datos en tiempo real, lo que permite un diagnóstico y tratamiento más precisos [41].
- **Automóviles:** los vehículos equipados con Edge AI pueden tomar decisiones de manera autónoma, como evitar colisiones y cambiar de carril, lo que mejora la seguridad en la carretera [41].
- **Sistemas de control de procesos:** los sistemas de control de procesos equipados con Edge AI pueden analizar y procesar datos en tiempo real, lo que permite una mejor toma de decisiones y una mayor eficiencia en la industria [41].

1.2.12 Hardware necesario para la aplicación de la visión artificial

Para aplicaciones de visión artificial, se requiere hardware específico para procesar y analizar grandes cantidades de datos de imágenes o video en tiempo real. A continuación, se describen algunos de los componentes de hardware necesarios para aplicaciones de visión artificial [42]:

1. Cámara: es el componente principal para capturar imágenes o video. Se puede utilizar diferentes tipos de cámaras, desde cámaras de teléfonos móviles hasta cámaras de alta resolución y cámaras térmicas especializadas.
2. Unidad de procesamiento: la unidad de procesamiento es el cerebro de la aplicación de visión artificial, y es responsable de realizar los cálculos necesarios para procesar las imágenes o el video. Las unidades de procesamiento pueden ser CPUs convencionales, pero las GPUs (Unidades de Procesamiento Gráfico) y las FPGAs (Matrices de Puertas Programables en Campo) son las más utilizadas para acelerar el procesamiento de imágenes.

3. Memoria: es esencial para almacenar los datos de imágenes o videos mientras se procesan. Se recomienda una cantidad adecuada para que la aplicación funcione sin problemas.
4. Dispositivos de almacenamiento: los dispositivos de almacenamiento son necesarios para almacenar las imágenes o el video capturado. Los dispositivos de almacenamiento más utilizados son los discos duros y las unidades de estado sólido (SSD).
5. Red: Para aplicaciones de visión artificial en tiempo real, la red es crucial para transmitir datos entre el hardware y los dispositivos de visualización.

1.2.13 Sistemas embebidos

Los sistemas embebidos se definen como un conjunto de componentes electrónicos integrados sobre una sola placa PCB para realizar funciones específicas, generalmente el procesamiento central se lleva a cabo por un microcontrolador o un microprocesador. Este tipo de sistemas se compone de tres componentes principales que son: hardware, software y firmware o sistema operativo. [43]

El hardware se refiere a los componentes físicos del sistema, como el procesador, la memoria RAM, el almacenamiento, los buses de comunicación y las interfaces de entrada/salida. El software es el conjunto de programas y algoritmos que permiten el funcionamiento del sistema embebido, puede ser codificación en C, C++ o Ensamblador que contienen las instrucciones que permiten la ejecución de las tareas específicas. El firmware es un software de bajo nivel que proporciona una capa de abstracción entre el hardware y el software, controlando y gestionando las operaciones básicas del sistema. [44]

A continuación, se presenta algunos ejemplos de sistemas embebidos, desde las populares tarjetas Arduino, BeagleBone y Raspberry Pi, hasta las potentes tarjetas NVIDIA Jetson, que impulsan aplicaciones de inteligencia artificial y visión por computadora. Aquí hay algunos

detalles sobre algunas de las tarjetas embebidas más populares tanto en la educación como en la industria:

Arduino: es una plataforma de prototipado electrónico muy popular que utiliza una tarjeta con un microcontrolador ATmega 328P de la familia ARM de ATmel. Estas tarjetas, como Arduino Uno o Arduino Mega, son capaces de interactuar con el entorno físico a través de sensores y actuadores, estas se programan utilizando el entorno de desarrollo integrado (IDE) de Arduino en lenguaje C y C++. [45]

BeagleBone: es un ordenador de código abierto cuyo propósito general es ser un dispositivo programable, se basa en un procesador ARM, almacenamiento, memorita RAM e interfaces de entrada y salida, su sistema operativo está basado en Linux y se puede programar en C y C++, C#, Java, Python, etc. [46]

Raspberry Pi: se considera comúnmente una computadora de placa única (SBC), y también se considera un sistema embebido por su tamaño compacto y capacidad de ejecutar aplicaciones específicas. Las tarjetas Raspberry Pi contienen un procesador, memoria, puertos de E/S y se pueden utilizar en una amplia gama de proyectos embebidos, como automatización del hogar, sistemas de vigilancia, servidores, entre otros. [47]

NVIDIA Jetson: Las tarjetas NVIDIA Jetson, como el Jetson Nano o Jetson TX2, están diseñadas específicamente para aplicaciones de inteligencia artificial y visión por computador. Estas tarjetas están equipadas con potentes GPU y procesadores, y se utilizan en proyectos que requieren capacidades de aprendizaje automático y procesamiento de imágenes en tiempo real. [48]

Tabla 2

Comparación entre sistemas embebidos, Fuente: Autoría Propia

Características	Arduino	Raspberry Pi	BeagleBone	NVIDIA Jetson
Tipo	Prototipado electrónico	Computadora de placa única	Computadora de placa única	Computadora de placa única
Procesador	ATmega 328P (ATmel)	Diversos modelos y marcas	Diversos modelos y marcas	NVIDIA GPU y procesador
Lenguaje de Programación	C, C++	Diversos lenguajes de programación (Python, C++, etc.)	Diversos lenguajes de programación (Python, C++, etc.)	Diversos lenguajes de programación (Python, C++, etc.)
Uso típico	Interacción con sensores y actuadores	Automatización del hogar, servidores de medios, sistemas de vigilancia	Proyectos embebidos con amplia conectividad	Proyectos de inteligencia artificial y visión por computadora
Capacidades	Baja potencia de cálculo y cómputo	Mayor potencia de cálculo y cómputo	Mayor potencia de cálculo y cómputo	Alto rendimiento en inteligencia artificial y procesamiento de imágenes
Aplicaciones destacadas	Proyectos de robótica y automatización simple	Automatización del hogar, servidores, proyectos multimedia	Automatización, sistemas embebidos interactivos	Proyectos de visión por computadora y aprendizaje automático
Conectividad	Puertos digitales y analógicos, UART, I2C, etc.	HDMI, USB, Ethernet, GPIO, etc.	HDMI, USB, Ethernet, GPIO, etc.	HDMI, USB, Ethernet, GPIO, etc.
Tamaño	Compacto y adecuado para prototipado	Compacto, pero más grande que Arduino	Compacto, pero más grande que Arduino	Compacto, especialmente Jetson Nano

En la tabla 3 se puede observar las características de cada sistema embebido que puede ser una solución para este trabajo de grado, dependerá de las especificaciones del proyecto para la selección de una de estas.

1.2.14 Software necesario para la aplicación de la visión artificial

Para aplicaciones de visión artificial, se requiere software especializado para procesar, analizar y comprender los datos de imágenes o video. A continuación, se describen algunos de los componentes de software necesarias para aplicaciones de visión artificial [49].

1. Bibliotecas de visión artificial: existen numerosas bibliotecas de visión artificial de código abierto, como OpenCV, TensorFlow, YOLO, PyTorch y Google Colab, que proporcionan herramientas para procesar imágenes y videos, detección de objetos, reconocimiento de patrones, aprendizaje profundo, etc.
2. Herramientas de etiquetado de datos: las herramientas de etiquetado de datos son esenciales para etiquetar y anotar imágenes o videos para entrenar modelos de aprendizajes automático. Algunas herramientas son LabelImg, RectLabel y CVAT.
3. Software de análisis de imágenes: es útil para extraer información de las imágenes o videos procesados, como la detección de objetos, la medición de características, la segmentación de imágenes. Algunos softwares: LabelImg, ImageJ, CellProfiler y Fiji.
4. Herramientas de programación: el conocimiento de lenguajes de programación como Python, C++ o MATLAB es esencial para desarrollar aplicaciones de visión artificial personalizadas y avanzadas.

1.2.14.1 Fases de funcionamiento de la visión artificial

El funcionamiento de la visión artificial se puede dividir en varias fases, que generalmente incluyen las siguientes [50]:

1. Adquisición de datos: en esta fase, se adquieren datos de imágenes o video utilizando una cámara u otros dispositivos de captura de imágenes. Los datos pueden ser capturados en tiempo real o pueden ser imágenes almacenadas previamente.

2. Preprocesamiento de datos: Una vez que se adquieren los datos, se deben preprocesar para mejorar la calidad y la claridad de las imágenes. Esto puede incluir la eliminación de ruido, la corrección de la distorsión, la mejora de la calidad de la imagen, la normalización del color y la mejora del contraste.
3. Segmentación de objetos: En esta fase, se separan los objetos de interés en la imagen del fondo. La segmentación puede ser realizada manualmente o utilizando técnicas automáticas de procesamiento de imágenes, como la segmentación basada en bordes, la segmentación basada en regiones y la segmentación basada en aprendizaje profundo.
4. Extracción de características: Una vez que se ha realizado la segmentación, se pueden extraer características relevantes de los objetos, como la forma, el tamaño, la textura, el color, la orientación y la posición. Esto se puede hacer mediante técnicas de procesamiento de imágenes, como la transformada de Fourier, la convolución, la detección de bordes y la extracción de características locales.
5. Reconocimiento y clasificación: En esta fase, se utiliza el conocimiento extraído de las características para reconocer y clasificar objetos en la imagen. Esto se puede hacer utilizando algoritmos de aprendizaje automático, como redes neuronales artificiales, SVM (máquinas de vectores de soporte), KNN (vecinos más cercanos) y clasificadores de árbol de decisión.
6. Toma de decisiones y acciones: Finalmente, se pueden tomar decisiones y acciones en función del resultado de la clasificación. Esto puede incluir el control de robots, la identificación de objetos defectuosos, el seguimiento de objetos en movimiento, el control de calidad y mucho más.

CAPÍTULO II

En este capítulo se presenta en profundidad el proceso de desarrollo del sistema, desde la descripción del proceso de selección de componentes electrónicos y visuales del sistema hasta concepción de la propuesta, además se profundiza en la programación y desarrollo de un sistema para la detección de cascos de seguridad aprovechando un lenguaje de alto nivel Python por ser de fácil aprendizaje y accesibilidad a librerías tanto de visión artificial para entrenamiento y ejecución de redes neuronales convolucionales como es el caso de YOLO.

2. MARCO METODOLÓGICO

2.1 Investigación

Se describe la información necesaria acerca del proceso de adquisición de imágenes, como lo son las etapas involucradas que siguen para analizar y extraer información útil de las imágenes o secuencias de imágenes. Así como la información necesaria para el desarrollo de la interfaz de usuario que será ejecutado sobre el sistema de embebido previamente seleccionado. Considerando que la investigación se desarrolla de principio a fin a lo largo del desarrollo del proyecto.

2.2 Materiales y métodos

Hardware

- Ordenador para programación
    - Modelo: Acer Nitro 5 AN515-53
    - Procesador: *Intel(R) Core(TM) i5-8300H CPU @ 2.30GHz*
    - Memoria instalada (RAM): *24,0 GB (23,8 GB usable)*
    - Tarjeta gráfica del sistema: Nvidia GTX 1050 4GB

  - Tipo de sistema: Windows 11 de 64 bits, procesador basado en x64

- Sistema de embebido
  - Modelo: Jetson Nano 2 GB.
  - Memoria ROM (MicroSD): 256 GB
  - Memoria RAM: 2 GB
- Cámara
  - Cámara Web
  - Resolución: 1080p
  - Ángulo de visión: 120°

Software

- Sistema operativo de 64 bits Windows 10
- Jetpack (Sistema Operativo de Jetson Nano)
- Anaconda (Entorno de programación)
- Spyder
- Python 3.11.2
- OpenCV
- Numpy
- YOLO
- Google Colab
- LabelImg

2.3 Diseño, nivel y tipo de investigación

El diseño, el nivel y el tipo de investigación son decisiones que se toman en base a los objetivos del proyecto, las preguntas planteadas y la naturaleza del problema a investigar. Cada

uno de estos elementos influye en la elección de métodos, técnicas y herramientas utilizadas en el proceso de investigación.

2.2.1 Diseño del programa

En el campo de la visión artificial, se pueden aplicar varias metodologías de investigación según los objetivos y las características específicas del proyecto. A continuación, se presentan algunas de las metodologías comunes utilizadas en la investigación de sistemas de visión artificial.

1. Diseño de algoritmos: Esta metodología se centra en el desarrollo de algoritmos y técnicas específicas para abordar problemas de visión artificial. Puede implicar el diseño y la implementación de algoritmos de detección, segmentación, reconocimiento de objetos, seguimiento de movimiento, entre otros.
2. Evaluación y benchmarking: En esta metodología, se lleva a cabo una evaluación sistemática y comparativa de los métodos y algoritmos existentes. Se utiliza como conjunto estándar y métricas de evaluación para comparar y medir el rendimiento de diferentes enfoques en tareas específicas, como la detección de objetos, la clasificación de imágenes, etc.
3. Análisis de rendimiento y optimización: esta metodología se centra en analizar y mejorar el rendimiento de los sistemas de visión artificial. Puede implicar el análisis de la velocidad de procesamiento, la precisión de la detección, el consumo de recursos computaciones, la optimización de algoritmos y la selección de parámetros óptimos.
4. Validación experimental: En esta metodología, se realizan experimentos y pruebas para validar y evaluar el rendimiento de un sistema de visión artificial en condiciones reales.

Esto puede incluir la recopilación de datos específicos, la implementación de sistemas en tiempo real, pruebas de robustez y comparaciones con métodos existentes.

2.2.2 Nivel de investigación

Para el desarrollo del presente proyecto se puede emplear un nivel de investigación descriptivo y aplicado.

- Nivel de investigación descriptivo: este nivel se centra en describir y caracterizar los fenómenos, en este caso, un objeto y su detección mediante visión artificial. La investigación descriptiva implicaría recopilar datos y analizar las características del objeto, como su forma, color, textura u otras propiedades visuales relevantes.
- Nivel de investigación aplicada: en este nivel, el enfoque principal es la aplicación práctica de técnicas de visión artificial para la detección del objeto. La investigación aplicada implicaría el desarrollo y la implementación de algoritmos y sistemas de visión artificial específicos para detectar y reconocer el objeto en imágenes o en tiempo real. Se pueden utilizar técnicas como detección de objetos, segmentación de imágenes, aprendizaje automático y clasificación para lograr una detección precisa y confiable del objeto.

2.2.3 Tipo de investigación

- Investigación cuantitativa: se utilizan métodos estadísticos para analizar y mediar el grado detección del sistema. Se pueden llevar a cabo estudios con un enfoque cuantitativo para determinar la precisión y la eficacia del sistema de visión en la detección de un objeto en imágenes o en tiempo real.
- Investigación experimental: en este tipo de investigación, se realizan experimentos controlados para evaluar y comparar diferentes enfoques o algoritmos de detección. Se pueden establecer condiciones específicas de iluminación, ángulos de visión y

escenarios para recopilar datos y evaluar el rendimiento del sistema. Esto implica la creación de conjuntos de datos de prueba y la ejecución de pruebas rigurosas para medir la sensibilidad y la especificidad del sistema.

- Investigación de campo: Se lleva a cabo la implementación y evaluación del sistema en un entorno real. Se pueden recopilar datos en tiempo real para evaluar la efectividad del sistema en condiciones reales y analizar los desafíos y las limitaciones encontradas.

2.2.4 Descripción de la investigación

Este estudio tiene como objetivo desarrollar un sistema que ejecute la rede neuronal YOLO para la detección e implementarlo en un sistema embebido para la supervisión del uso de cascos de seguridad en empresas o instituciones que utilicen cascos de seguridad.

Este estudio consta de nueve etapas que se describen a continuación:

Etapa 1: Esta etapa involucra actividades de recopilación de información para obtener datos sobre el objeto a detectar, los posibles algoritmos a utilizar para el desarrollo del sistema y las posibles tarjetas de desarrollo embebidas que permitan ejecutar el sistema.

Etapa 2: Incluye la realización del algoritmo para el sistema de detección. Considerando el equipo en el que se desarrolla (Ordenador) y en el que se ejecuta (Sistema de embebido) tendrá las siguientes consideraciones.

- Sistema embebido

Selección del sistema embebido que mejor se adecue a los requerimientos y características del proyecto.

Instalación del sistema operativo Jetpack en la placa embebida Jetson Nano y también el entorno de programación Spyder. Además de las librerías OpenCV y numpy.

- Ordenador para programación

Instalación de la plataforma Anaconda y uso del entorno de programación Spyder. Además de las librerías OpenCV y numpy.

Etapa 3: Creación de un dataset de las imágenes en el ordenador para el análisis de las características del objeto.

Etapa 4: Etiquetado de las imágenes en donde se resalte el objeto de estudio con la finalidad de obtener las coordenadas que delimitan el objeto a detectar. Este resultado obtenido se carga en un almacenamiento en la nube.

Etapa 5: Entrenamiento de la red neuronal YOLO con la base de datos almacenados en la nube, para la obtención de los pesos que se utilizaran para la detección del objeto.

Etapa 6: Realización del programa para el sistema de detección y la interfaz de usuario.

Etapa 7: Validación, análisis de resultados y corrección del proyecto.

2.4 Selección de sistema embebido

En el desarrollo del proyecto, se ha seleccionado la NVIDIA Jetson Nano de 2GB como una herramienta fundamental para llevar a cabo la investigación en el campo de la visión artificial. Esta tarjeta de desarrollo altamente eficiente y potente ha sido elegida para realizar el análisis de imágenes y el procesamiento de visión en tiempo real. Con su capacidad de procesamiento y su memoria RAM de 2GB, la Jetson Nano permite ejecutar algoritmos y aplicaciones de visión por computadora de manera precisa y rápida.

Además, su versatilidad y las interfaces de E/S disponibles permiten la conexión de cámaras y otros dispositivos necesarios para recolectar y procesar los datos visuales. Con la Jetson Nano de 2GB como parte integral del proyecto, se busca explorar nuevas técnicas y enfoques en el

campo de la visión artificial, con la aspiración de obtener resultados significativos que contribuyan al avance de esta área de estudio.

Tabla 3

Tarjeta embebida seleccionada para el sistema

	Características **NVIDIA Jetson 2GB**
Tipo	Computadora de placa única
Procesador	NVIDIA GPU y procesador
Lenguaje de Programación	Diversos lenguajes de programación (Python, C++, etc.)
Uso típico	Proyectos de inteligencia artificial y visión por computadora
Capacidades	Alto rendimiento en inteligencia artificial y procesamiento de imágenes
Aplicaciones destacadas	Proyectos de visión por computadora y aprendizaje automático
Conectividad	HDMI, USB, Ethernet, GPIO, etc.
Tamaño	Compacto, especialmente Jetson Nano

La elección de la NVIDIA Jetson Nano de 2GB se basa en sus especificaciones técnicas destacadas, que la convierten en una opción idónea para el proyecto de visión artificial. Esta tarjeta cuenta con un procesador ARM Cortex-A57 de cuatro núcleos, que ofrece un rendimiento eficiente para ejecutar aplicaciones de visión artificial. Además, sus 2GB de memoria RAM permiten procesar grandes cantidades de datos visuales de manera rápida.

La Jetson Nano cuenta con una GPU NVIDIA Maxwell de 128 núcleos, dando una capacidad de procesamiento gráfico de alto rendimiento para acelerar la detección de objetos y la ejecución de algoritmos complejos. Esta GPU acelerada permite ejecutar algoritmos de visión por computadora de forma eficiente, aprovechando su potencia de cálculo paralelo para realizar tareas como detección de objetos, reconocimiento facial y seguimiento de movimiento con mayor velocidad y precisión.

Asimismo, las interfaces de E/S disponibles, como puertos USB, HDMI, Ethernet y GPIO, ofrecen una amplia conectividad para la integración con otros dispositivos y sensores requeridos en el proyecto. Gracias a estas especificaciones técnicas, la Jetson Nano de 2GB brinda el equilibrio adecuado entre potencia de cálculo y eficiencia energética, lo que la convierte en la elección ideal para abordar los desafíos de la visión artificial.

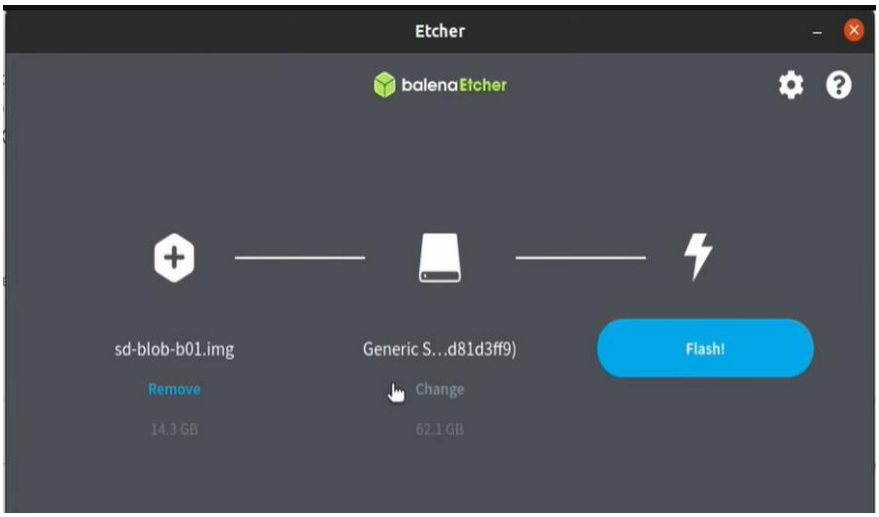
La combinación del procesador ARM y la GPU NVIDIA en la Jetson Nano de 2GB ofrece un equilibrio ideal entre tareas de cómputo general y procesamiento gráfico, lo que la convierte en una opción ideal para aplicaciones de visión artificial en sistemas embebidos. Esta potencia de cálculo permite realizar análisis y detección de objetos en tiempo real y llevar a cabo tareas complejas de procesamiento visual con gran eficiencia y precisión.

2.5 Instalación de Software sobre el sistema embebido

Para instalar el sistema operativo dentro de Jetson Nano de 2GB es necesario descargar el sistema operativo en formato IMG y cargarlo sobre una memoria SD en el software de balenaEtcher.

Figura 14

Interfaz de Balena Etcher para flashear sistemas operativos en tarjeta SD, Fuente: Autoria Propia.



En la figura 12 se observa en la interfaz de BalenaEtcher ya cargada la imagen del sistema operativo en formato .IMG lista para ser flasheada en la microSD que se va a utilizar como memoria ROM en el sistema embebido. El sistema operativo que se instala en la Jetson Nano de 2GB viene con Python 3.0 instalado por defecto, pero es necesario actualizar para tener la última versión de Python o en su defecto instalar la misma versión que se esta usando en la PC en la que se está desarrollando el software.

Una vez cargada la imagen .IMG del sistema operativo ya podemos encender la placa de desarrollo Jetson y poner en marcha el sistema Operativo para la posterior instalación de un entorno de programación.

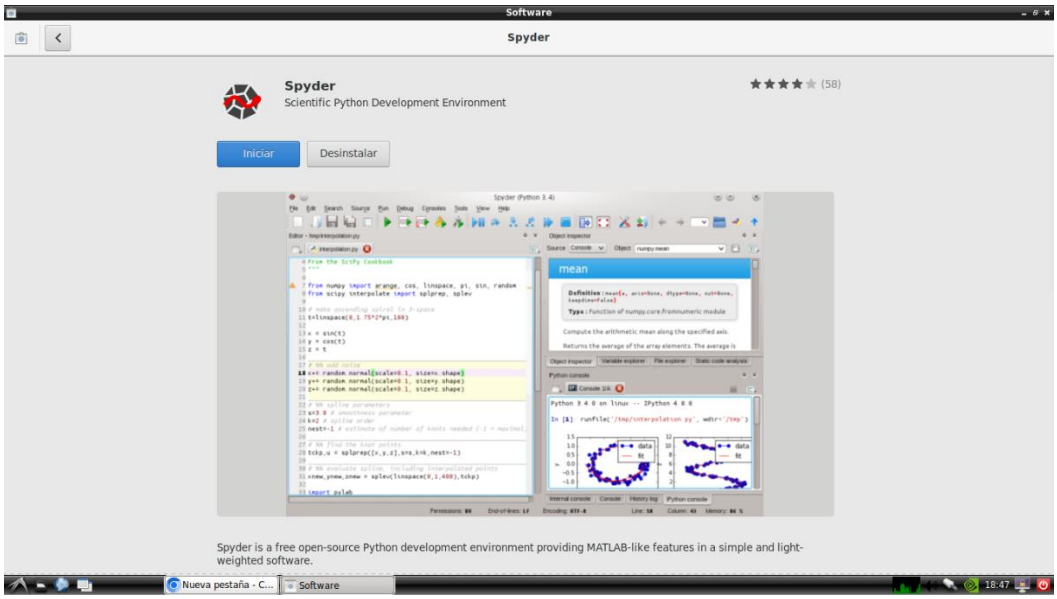
2.5.1 Instalación de Spyder dentro del Sistema Embebido

Para ejecutar cualquier código dentro de la tarjeta Jetson se puede usar la interfaz integrada de pythonn o usar un entorno que sea compatible con el sistema operativo. En este caso se usa el entorno de programación Spyder, para evitar incompatibilidad en el programa desarrollado en la PC al momento de ejecutarse sobre el sistema embebido. Para instalar Spyder dentro de esta placa, con las librerías e instancias correctas se debe seguir los siguientes pasos:

1. Abrir el buscador de aplicaciones dentro de Jetson Nano
2. Buscar la aplicación llamada Software y en la barra de buscar escribir “Spyder”
3. Aparecerá el entorno listo para instalarse como se muestra en la figura 14
4. Presionar Instalar para que descargue e instale el entorno de desarrollo.

Figura 15

Entorno de desarrollo Spyder dentro del sistema embebido Jetson nano 2GB. Fuente: Autoria propia.



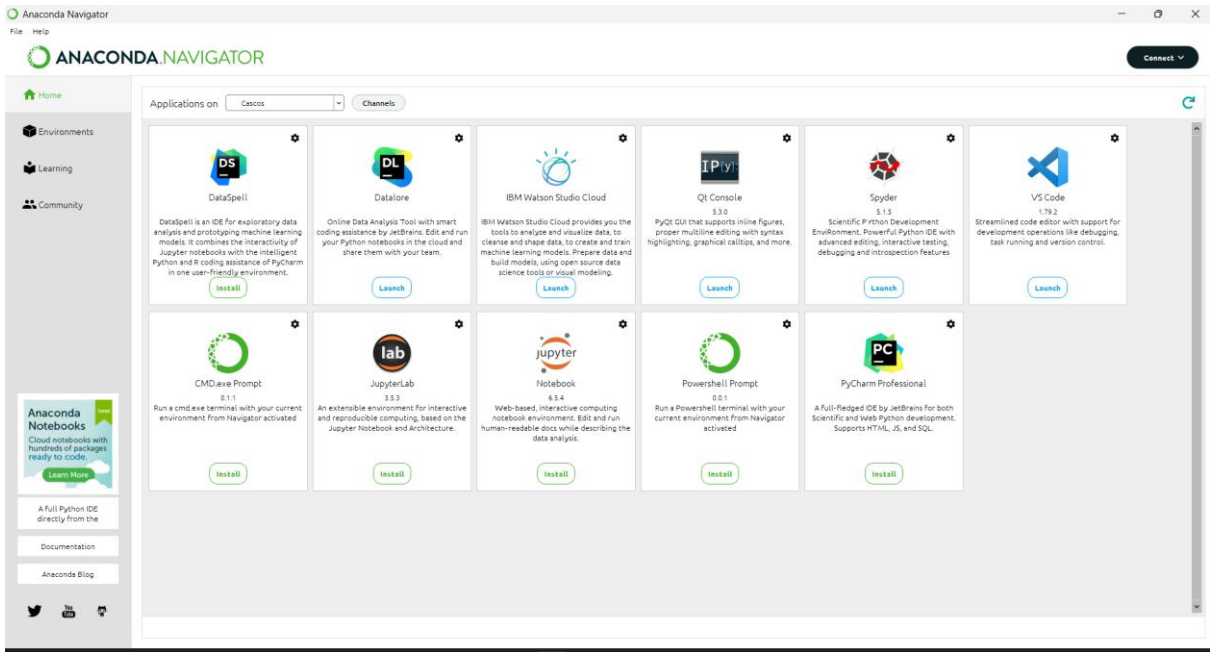
En la figura 14 se observa que Spyder se encuentra correctamente instalado y listo para su ejecución, la versión que existe para la Jetson Nano 2GB no es la ultima que han lanzado los desarrolladores de Spyder pero es compatible con las distribuciones de PC, es decir es compatible con cualquier versión que se use en la computadora con sistema Windows.

2.6 Instalación de Anaconda en la PC

Anaconda es una distribución de los lenguajes de programación Python y R para computación científica (ciencia de datos, aplicaciones de Machine Learning, procesamiento de datos a gran escala, análisis predictivo, etc.). [51]

Figura 16

Interfaz de Anaconda para instalar paquetes de Python.



En la figura 15 se puede observar la interfaz de Anaconda con los entornos de desarrollo más populares que se usan para programar en Python.

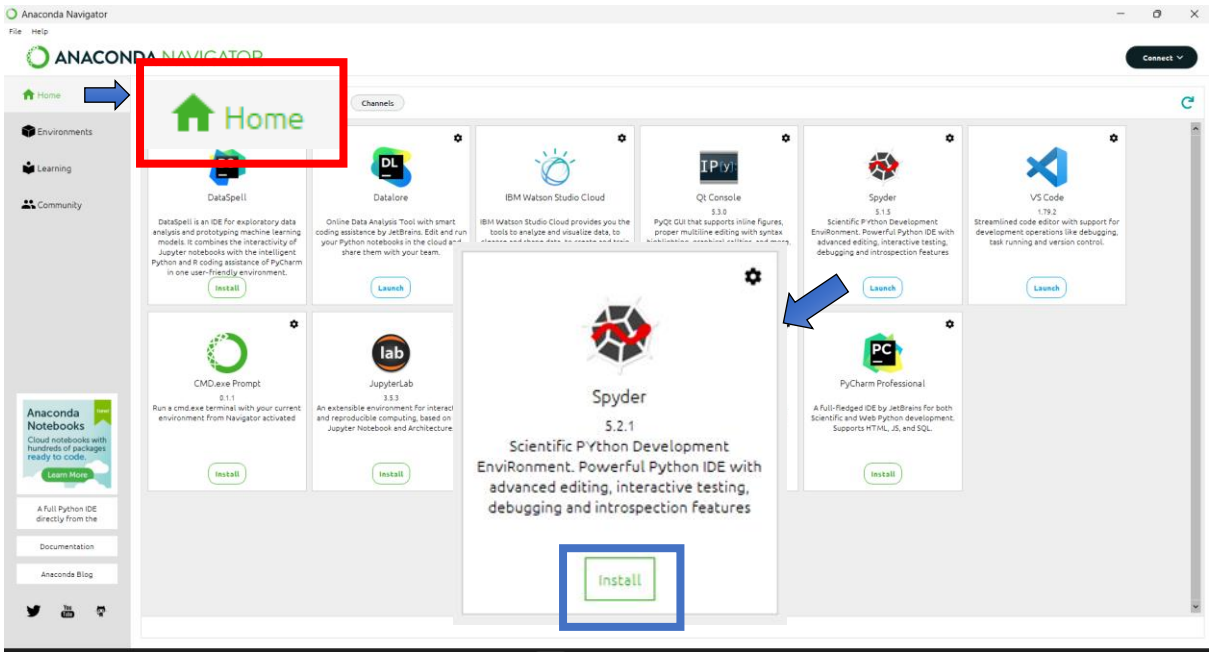
2.6.1 Instalación de Spyder en PC

Spyder es un potente entorno de programación en Python, es un entorno de desarrollo integrado y multiplataforma de código abierto (IDE) para programación científica en el lenguaje Python. Este IDE se liberó bajo la licencia de MIT. Spyder es extensible con complementos. Incluye soporte de herramientas interactivas para la inspección de datos e incorpora controles de calidad específicos de Python e instrumentos como Pyflakes, Pylint y Rope, además que se usa para ejecutar código de Python con las librerías OpenCV, numpy, etc. [52]

Para instalar Spyder en Anaconda como se muestra en la figura 16 se debe dirigir a “Home” en la parte superior izquierda de su pantalla y luego se busca el icono de Spyder y se selecciona la opción de instalar. En este caso se instaló la versión 5.1.5 de Spyder.

Figura 17

Instalación de Spyder en la PC.



2. 7 Desarrollo del software para el sistema de detección

En este apartado se desarrolla las etapas que se llevan a cabo para obtener la aplicación que se ejecutará en el sistema embebido seleccionado, teniendo en cuenta que debe ser sencilla de usar y que no consuma muchos recursos.

2.7.1 Creación del dataset

El proceso de creación de un dataset de imágenes para usar en el proceso de entrenamiento de una CNN es importante para obtener todas las características del objeto que se vaya a detectar.

En este sistema los objetos a detectar son los cascos de seguridad, las características que se consideraron para el análisis son las siguientes:

1. Detección del casco: identificar si un casco de seguridad está presente en la imagen o no.

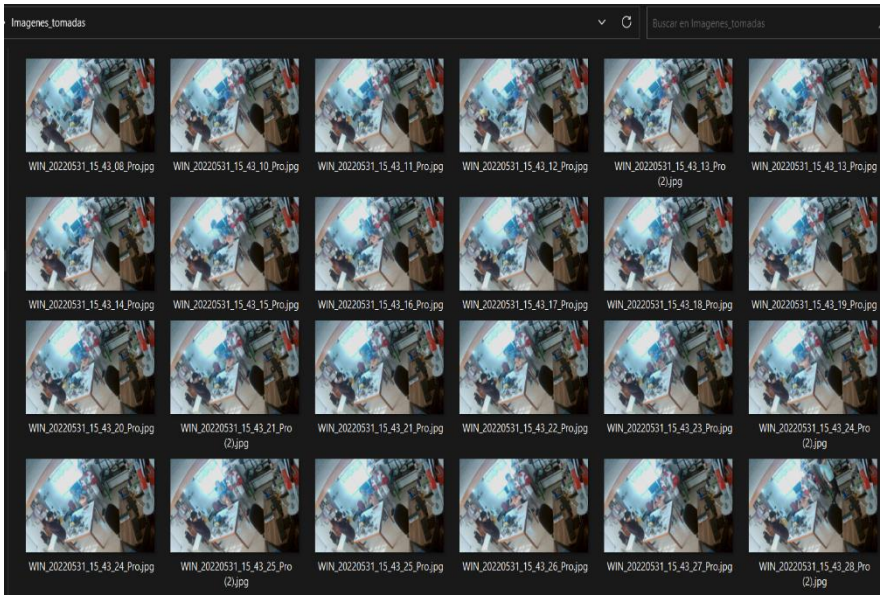
2. Posición y orientación: determinar la posición y orientación del casco dentro de la imagen, mostrando una etiqueta del objeto en tiempo real.

2.7.1.1 Captura de fotografías

Con el objetivo de crear un dataset que contenga las imágenes con los cascos de seguridad en la figura 16 se puede observar la captura de las fotografías que serán usadas, tanto para el entrenamiento como para la validación del sistema. El total de fotografías capturadas para el entrenamiento de la red fue de 6343, la captura de las mismas se realizó en una oficina, en un taller y además se agregó fotos del torso de una persona usando el casco y sin usar el cascos, para obtener mejores resultados en el caso de detectar con una webcam frontal.

Figura 18

Recopilación de imagenes para crear el dataset.



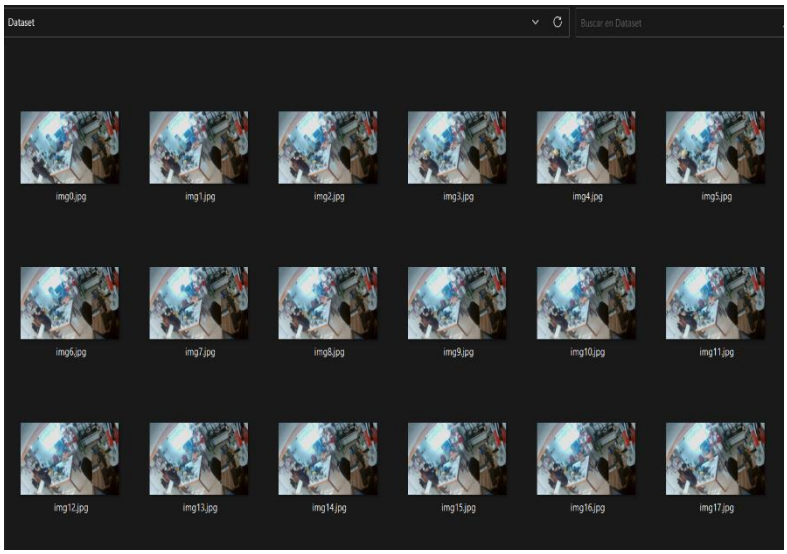
2.7.1.2 Ordenamiento de fotografías

En el código del anexo 1 se observa el código de ordenamiento que no permite tanto cambiar el nombre de las imágenes como cambiarles el nombre en un orden determinado para tener un orden en los datos de entrada para la red neuronal. El ordenamiento de las fotografías es

necesario para tener todas las imágenes en secuencia y revisar el entrenamiento cuando se este ejecutando.

Figura 19

Cambio de nombre y ordenamiento de las imágenes de dataset



En la figura 17 se observa que las imágenes fueron ordenas desde la img0 hasta la img17, por lo tanto, se comprueba que el mismo proceso se hará hasta la imagen 6343. La finalidad de cambiar los nombres y ordenar las fotografías sirve para poder comprobar que tanto el entrenamiento como la validación de la red se realice correctamente, al poder verificar que no se ha tenido errores en el proceso de etiquetado, además el programa del anexo 1 permite cambiar la extensión del archivo para que sea compatible con el programa de etiquetado.

2.7.1.3 Etiquetado de dataset

Para etiquetar las imágenes se usó un software de código abierto llamado LabelImg programado en Python y que usa la librería PyQT5 para su interfaz gráfica. Este programa permite etiquetar fotografías en formatos png, jpg, ico, gif, etc.

En la figura 19 se observa el logo del programa LabelImg que permite etiquetar las imágenes del dataset. A continuación, en la interfaz del programa se debe buscar la ubicación del dataset dentro de la computadora.

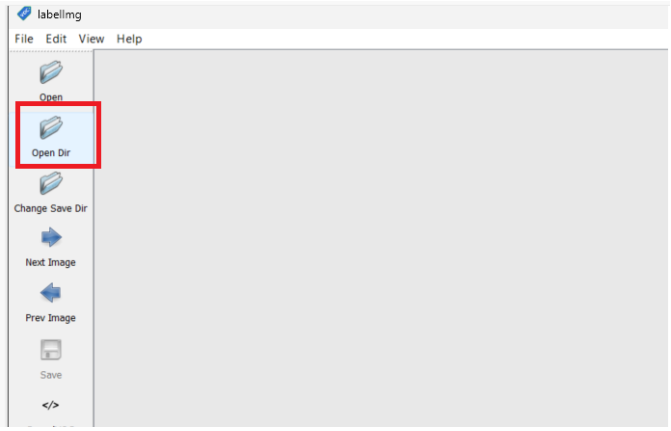
Figura 20

Icono del programa LabelImg.

En la Figura 20, se muestra el botón Open Dir, que permite buscar el directorio donde se encuentra guardadas las imágenes del dataset ya ordenadas y listas para ser etiquetadas.

Figura 21

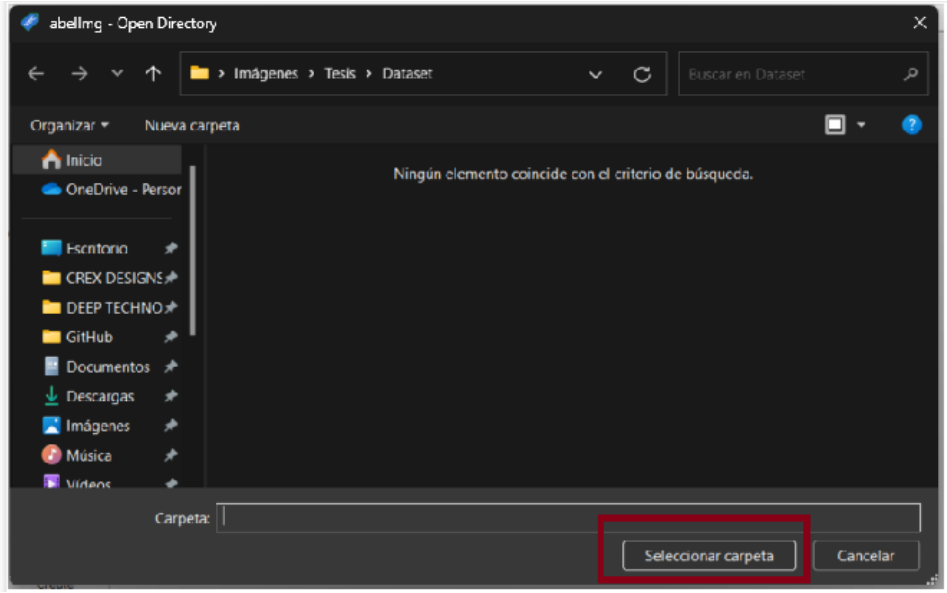
Botón de busqueda del directorio del dataset.



En la Figura 21, se observa el directorio Imágenes/Tesis/Dataset, esta es la ubicación para el dataset, lo siguiente es dar clic en el botón “Seleccionar carpeta” para que se guarde la carpeta de donde se estarán tomando las imágenes que van a ser etiquetadas.

Figura 22

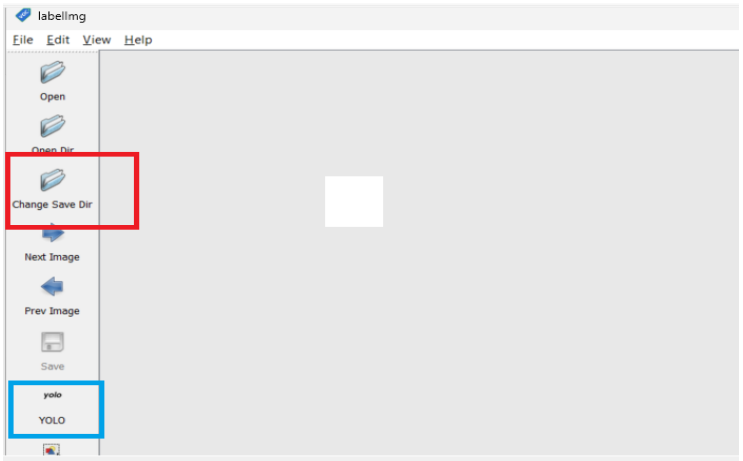
Búsqueda del dataset dentro del PC.



Una vez abierto el dataset se procede a darle ubicación a los archivos txt y además configurar el formato YOLO en el que trabaja la red.

Figura 23

Botón de dirección de guardado de etiquetas.

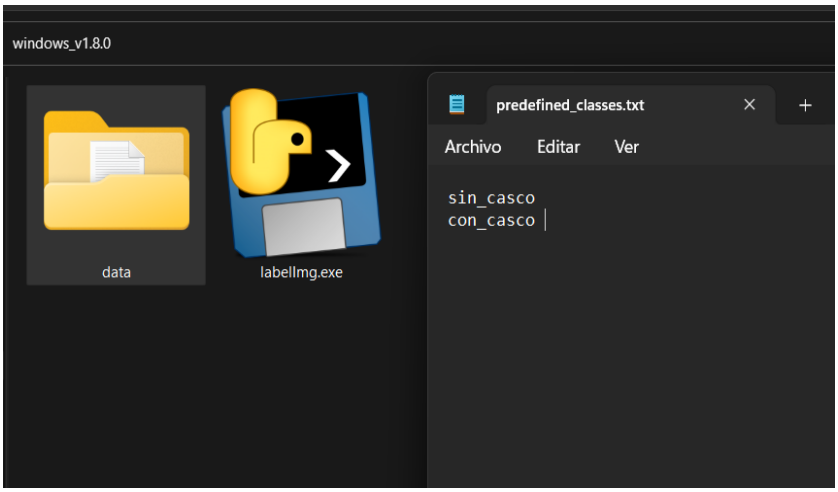


Para empezar a etiquetar las imágenes se deben establecer los objetos que se van a detectar, en el campo de la visión artificial se les llama clases y se las puede nombrar con el objeto que se quiere detectar. En este caso para el sistema se crean dos clases: “sin casco” y “con casco”.

Para crear las clases que se van a usar se debe editar un archivo de texto que viene junto al programa labelImg en la misma carpeta del archivo ejecutable, en la carpeta data en el directorio windows_v1.8.0\data, una vez dentro como se muestra en la Figura 23, se debe abrir el archivo "predefined_classes.txt", y escribir las clases que se necesiten con un salto de línea.

Figura 24

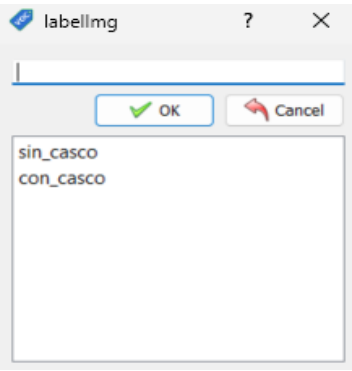
Definición de clases a detectar



Una vez realizado todo eso, se puede cerrar "predefined_classes.txt" y abrir el programa labelImg para revisar que se hayan creado las clases para empezar el etiquetado.

Figura 25

Clases en labelImg para el etiquetado

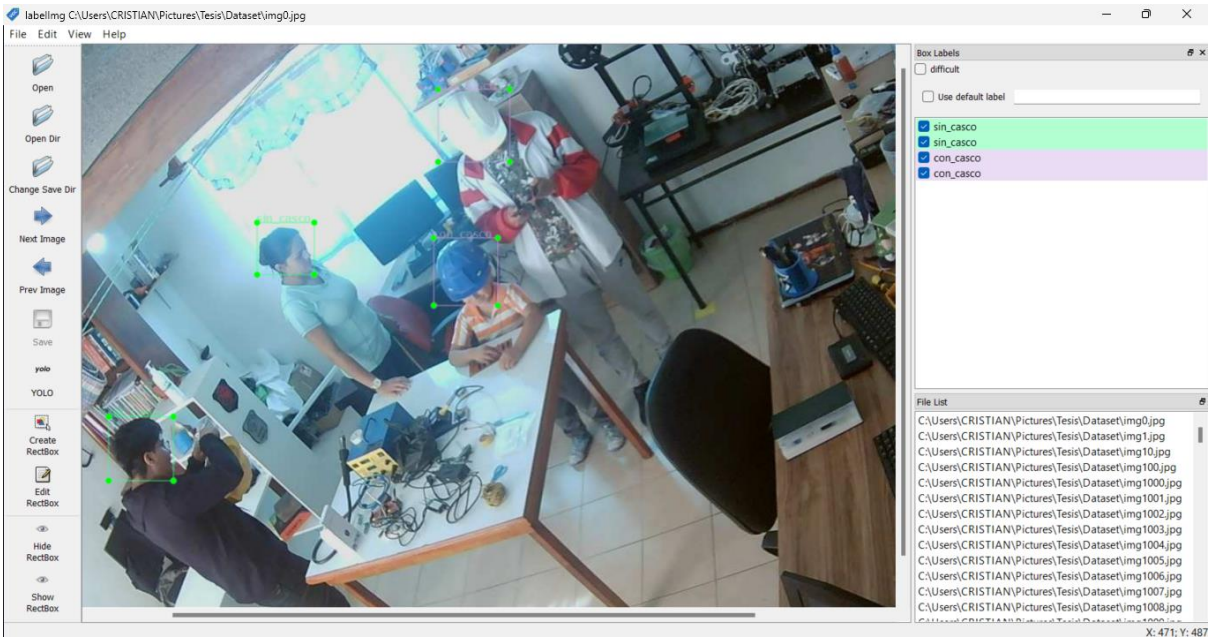


En la Figura 24, dentro del programa labelImg se observa que las clases creadas son las correctas y se puede etiquetar cada clase según corresponda con la imagen que se esté trabajando.

En la Figura 26, se muestra la interfaz completa del programa con la primera imagen del dataset etiquetada correctamente, es importante etiquetar correctamente todas las imágenes del dataset para evitar errores tanto en el entrenamiento como en la validación del sistema.

Figura 26

Etiquetado de imágenes en labelImg



Se hace énfasis en que el etiquetado sea correcto porque el sistema puede detectar los objetos de manera incorrecta o simplemente no detectar nada. Como se muestra en la Figura 26, en cada etiqueta debe contenerse el objeto que se busca detectar y una recomendación es que para crear una etiqueta almenos el 50% de la etiqueta debe encerrar al objeto, pero esto en el caso extremo, es mejor encerrar todo el objeto para evitar errores.

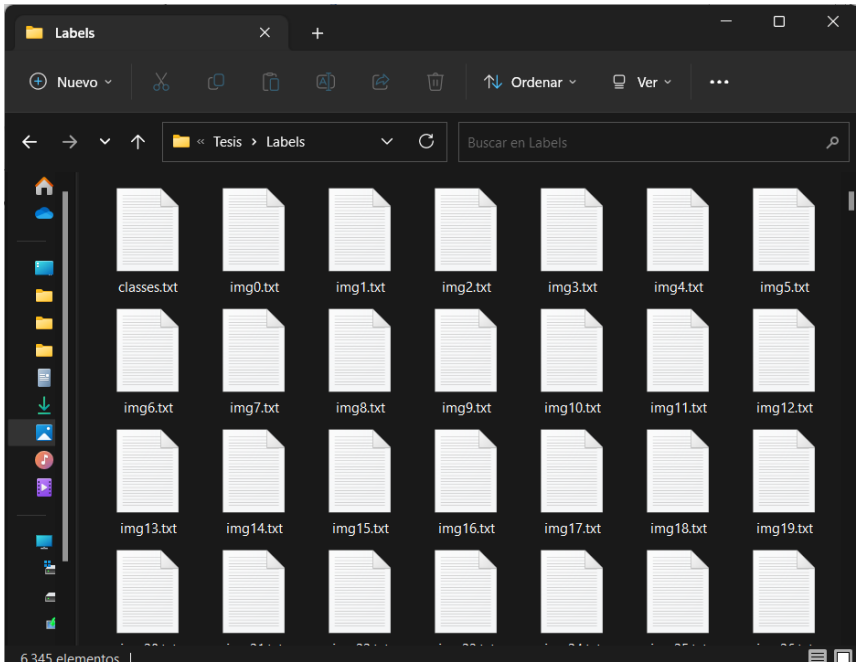
Figura 27

Etiquetado correcto de las imágenes

Una vez se hayan etiquetado todas las imágenes del dataset, en la Figura 27 se puede observar en la carpeta que se selecciona como Save Dir, que se han creado archivos .txt que contienen las coordenadas de cada una de las etiquetas en cada imagen con el mismo nombre de la imagen, esto crea una relación entre la imagen y sus etiquetas que más adelante se usa para crear la carpeta de entrenamiento que se subirá a la nube.

Figura 28

Archivos .txt generados con labelImg



2.7.2 Entrenamiento de red neuronal YOLO

Para poder entrenar la red neuronal YOLO, para detectar los cascos de seguridad en tiempo real, primero se debe crear los directorios que se van a subir en Google Colab .

El primer directorio contiene los archivos .txt con información del etiquetado donde están los labels de cada clase definida.

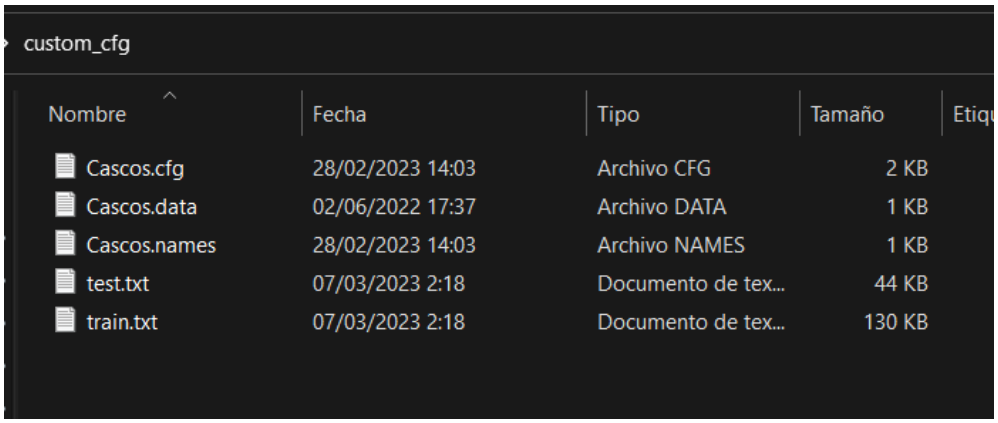
En la Figura 29 se muestra el segundo directorio contiene los archivos:

- Pesos Descargados para realizar transfer learning
- Archivo de Cascos.cfg, el cual contiene la configuración de nuestra red.
- Archivo de Cascos.data, el cual contiene las direcciones de nuestros archivos a usar en el entrenamiento.
- Archivo de Cascos.names, el mismo que contiene el nombre de las clases u objetos a detectar.
- Archivo train.txt y tests.txt, los mismos que contienen los paths(rutas) de las imágenes a utilizar en el entrenamiento y pruebas específicamente.

El archivo de pesos pre-entrenados, el cual nos permite aplicar transfer learning y de esta manera nuestro algoritmo de detección de objetos aprenda más rápido, debido a que no empieza aprender desde 0, se lo puede descargar desde darknet53.conv.74.

Figura 29

Archivos de configuración de red neuronal YOLO V3



2.7.2.1 Creación de archivos necesarios para el entrenamiento

1. Creación del conjunto de entrenamiento y prueba

Para realizar el entrenamiento se necesita dividir el conjunto de datos en: un conjunto de datos de entrenamiento y un conjunto de datos de prueba. Para ello, se crea dos archivos en formato .txt, en los cuales se almacena las direcciones de las imágenes que se utilizará para entrenar y probar el modelo.

La creación de estos archivos, se lo realiza mediante el siguiente código:

```
# importamos librerias
import numpy as np
import matplotlib as plt
import glob, os

# obtenemos las direcciones  de las imagenes en nuestro dataset
paths =glob.glob("custom_dataset/*.png")
#creamos nuestros archivos de almacenamiento
file_train = open("train.txt","w")
file_test = open("test.txt","w")
```

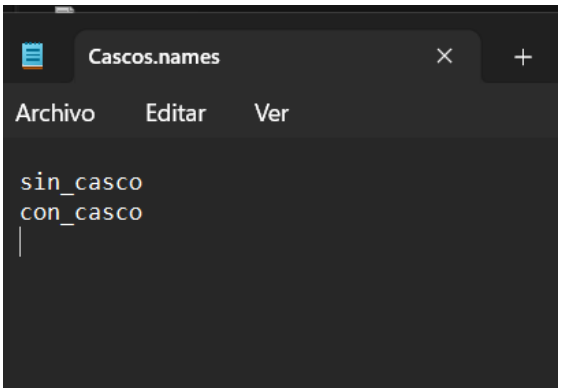
```
# configuramos para obtener un 20% de datos de entrenamiento
num_test = int(len(paths)/(0.2 *len(paths)))
con = 0
for path in paths:

    if (con == num_test):
        path=path.replace('\\','/')
        file_test.write(path+ '\n')
        con = 0
    else:
        path=path.replace('\\','/')
        file_train.write(path + '\n')

    con +=1

file_test.close()
file_train.close()
```

2. Creación del Cascos.names

A continuación como se observa en la Figura 30, se crea un archivo de texto Cascos.names, el cual contiene las clases o nombres de los objetos a detectar, en esta ocasión, lleva el nombre de sin_casco, con_casco, dado que sólo se va a detectar dos objetos.

Figura 30

Archivo .names con los nombres de las clases



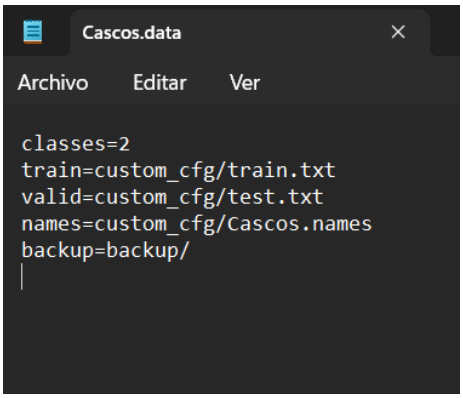
3. Creación del Cascos.data

El archivo Cascos.data contiene la información con la cual se va a entrenar la red, en primera instancia el número de clases o número de objetos a detectar, segundo: la dirección del archivo de texto que contiene los paths o rutas de las imágenes a usar en el entrenamiento, tercero: la dirección del archivo de texto que contiene los paths o rutas de las imágenes a usar en la validación o testing de la red, cuarto: el archivo de texto .names el cual contiene los nombres o clases de los objetos a detectar y quinto: la dirección de la carpeta donde se almacenarán los pesos de la red cada 4000 épocas de entrenamiento.

La creación de este archivo debe quedar como la Figura 31 de a continuación:

Figura 31

Archivo .data con la configuración de directorios



4. Creación del archivo de configuración Cascos.cfg

Para crear el archivo de configuracion de la red y entrenar el detector de objetos personalizado se debe copiar el contenido del archivo yolov3_tiny.cfg en un nuevo archivo de texto con un nombre personalizado con extensión .cfg, en este caso Cascos.cfg.

En este archivo se realiza las siguientes ediciones que se muestran en las Figuras 32, 33 y 34:

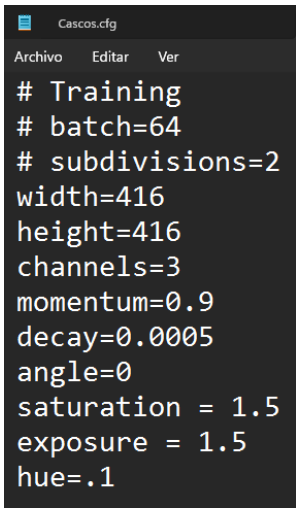
Cambiar la línea 6 lote de entrenamiento a batch=64, que especifica la cantidad de imágenes en lote que se va a cargar.

Cambiar la línea 7 subdivisiones del lote a subdivisions=8, lo que especifica que el lote de entrenamiento será dividido en 8 partes de 8 imágenes, al culminar de procesar un lote se ajusta los parámetros de la red y se guarda su configuración.

Cambiar la línea 8y 9, alto y ancho de la imagen que recibirá la red específicamente, width=416, height=416 o cualquier múltiplo de 32 siendo el tamaño mínimo aceptable 32*32px.

Figura 32

Archivo Cascos.cfg modificable



Se cambia la línea classes=80 con el número de objetos a detectar, en este caso 2 en cada una de las capas de YOLO.

Figura 33

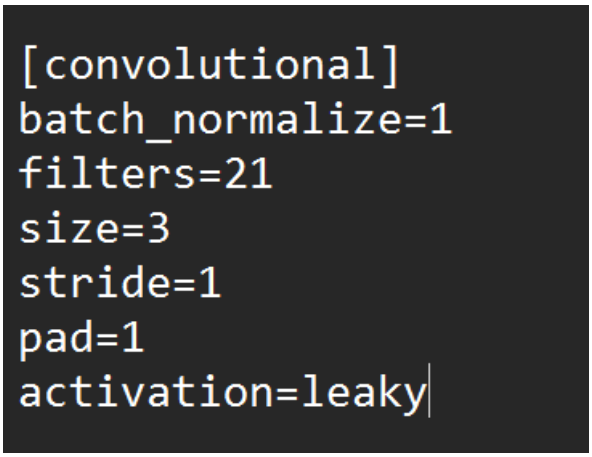
Archivo Casco.cfg cambio de clases

```
[yolo]
mask = 3,4,5
anchors = 0.83,1.34, 1.04,2.01, 1.57,1.49, 1.83,3.64, 2.26,1.95, 3.24
classes=2
```

De igual manera se cambia el número de filtros [filters=256] a filtros=(# de clases + 5) 3, en este caso (2+5)3=21 filtros.

Figura 34

Archivo Cascos.cfg para cambio de numero de filtros

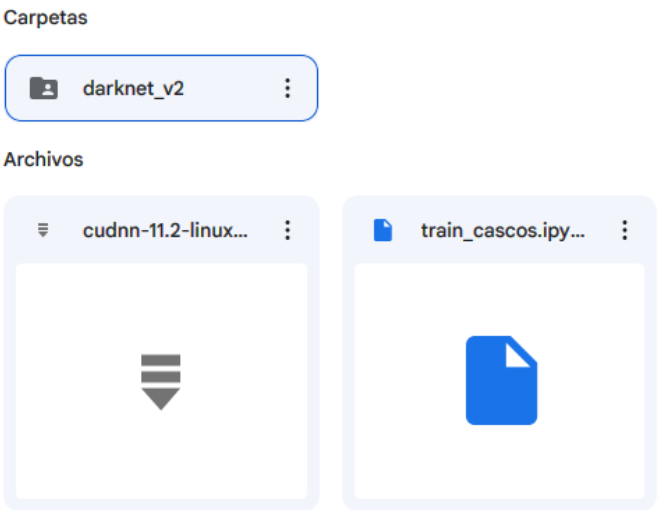


5. Ejecución del entrenamiento en Google Colaboratory

Se suben los archivos a Google drive en la carpeta raíz de la nube como se muestra en la Figura 35 y desde Google colab se ejecutan las líneas a ejecutarse.

Figura 35

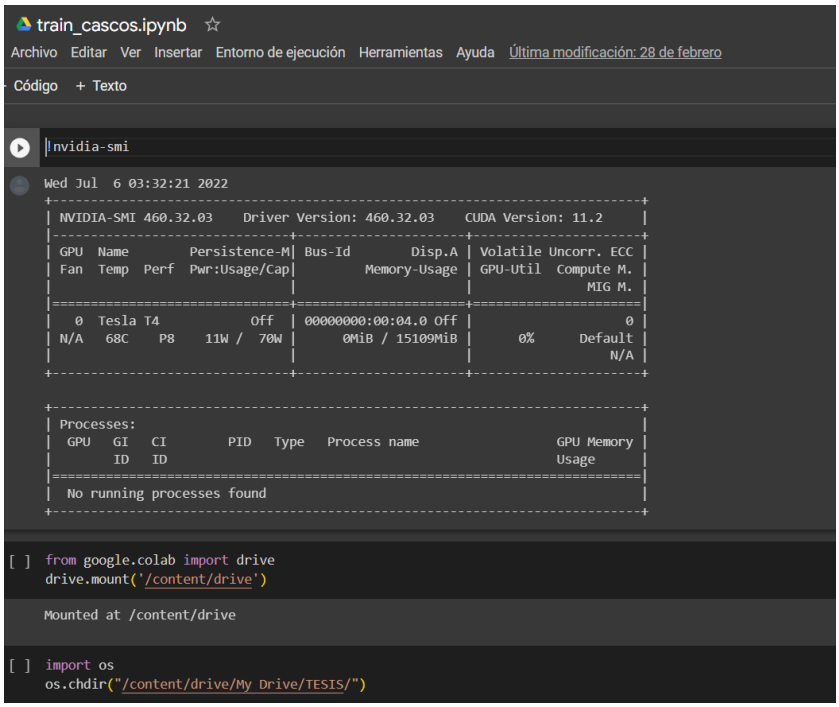
Archivos subidos a Google Drive



En Google drive deben estar todos estos archivos para poder ejecutar el entrenamiento de la red neuronal YOLO. Ahora con todo listo se procede a abrir Google Colab para ejecutar el código de entrenamiento de la red como se muestra en la Figura 36.

Figura 36

Vista de Google Colab para entrenar la red

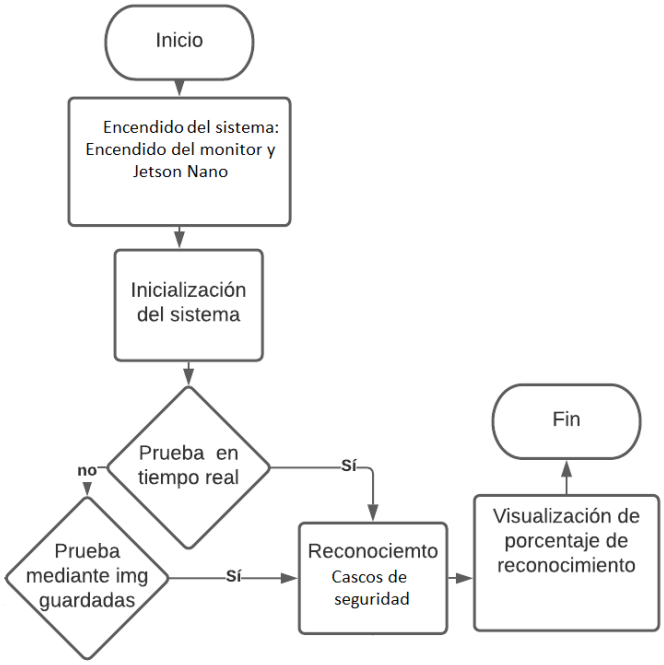


2.8 Algoritmo de programación para la detección de objetos

Una vez terminado el entrenamiento se configura la red con los pesos que arroja el entrenamiento y se ejecuta la red para comprobar que esté funcionando correctamente.

Figura 37.

Diagrama de bloques del programa de detección



El sistema de detección se basa en el diagrama de flujo de la figura y representa como se va a ejecutar el sistema una vez arranque.

2.9 Interfaz de usuario

La interfaz de usuario debe ser sencilla de entender e intuitiva para que su uso sea sencillo, se debe limitar a tener los botones necesarios para la ejecución del sistema sin que el usuario llegue a tener confusión.

Para el desarrollo de la interfaz se usa una librería integrada en Python llamada Tkinter que sirve para la creación y desarrollo de aplicaciones de escritorio. Esta librería facilita el posicionamiento y desarrollo de una interfaz gráfica de escritorio con Python. En la documentación de Python se describe la tkinter como la interfaz por defecto de Python para el kit de herramientas de GUI Tk, estando disponible en todas las distribuciones de Unix, así como en sistemas Windows y Mac OS.

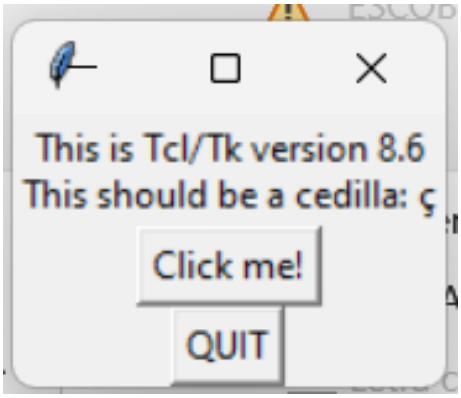
Por lo general a partir de Python 2.0 la librería de tkinter viene instalada por defecto pero puede haber casos en que no, para esto es necesario revisar la versión de tkinter y comprobar si está instalada y en que versión se encuentra, para actualizar la librería en caso de requerirse. Para revisar la versión de Tkinter que se encuentra instalada en el equipo se escribe en la línea de comandos de Anaconda:

```
python -m tkinter
```

Con este comando se ejecuta una ventana de tkinter simple donde muestra la interfaz de tkinter instalada en el sistema. También muestra la versión de Tcl/Tk para buscar la documentación especifica dependiendo de la versión instalada en el sistema.

Figura 38

Ventana de la versión de Tkinter



La librería de Tkinter proporciona una manera sencilla de crear interfaces gráficas simplemente codificando en base a widgets que van a ir componiendo poco a poco la interfaz. El widget principal que se usa es Tk o raíz donde se van a colocar el resto de los widgets de la interfaz gráfica, como botones, cuadros de texto, cuadros para mostrar la cámara, etc.

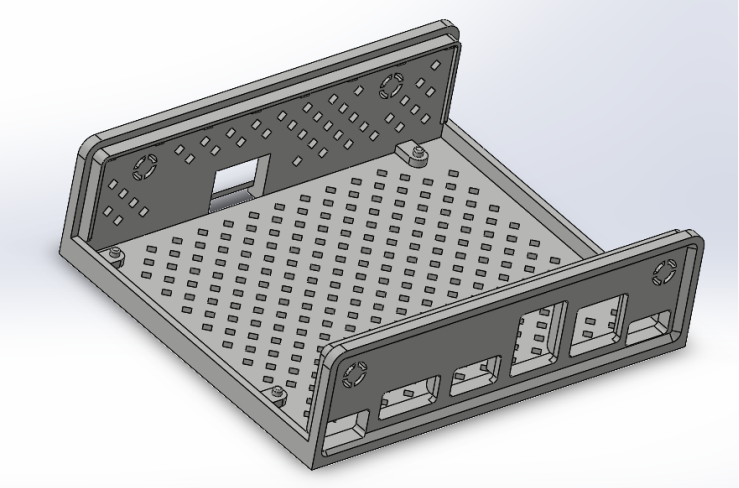
2.10 Diseño CAD de carcasa para Jetson Nano 2GB

El prototipo debe ser funcional y para esto se diseña un modelo CAD que sirva como carcasa para ensamblar tanto el sistema embebido como los demás componentes de alimentación y cámara.

En la Figura 39, se muestra la parte inferior de la carcasa que contiene la placa del sistema embebido Jetson Nano 2GB con sus respectivos puertos para la fuente poder, puertos USB, puertos Ethernet y además tiene perforaciones que servirán para la ventilación de la placa.

Figura 39

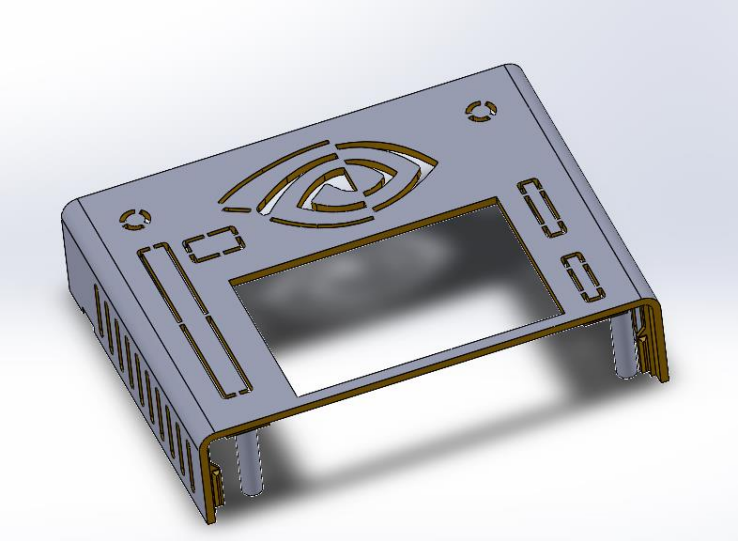
Parte superior de carcaza



En la Figura 40, se muestra la parte superior de la carcasa que protege la placa del sistema embebido Jetson Nano 2GB con una salida para el disipador de calor y algunas entradas de aire, además de una ranura que permite la interacción con los módulos de entradas y salidas que tiene esta placa.

Figura 40

Parte inferior de carcaza



CAPÍTULO III

3. RESULTADOS

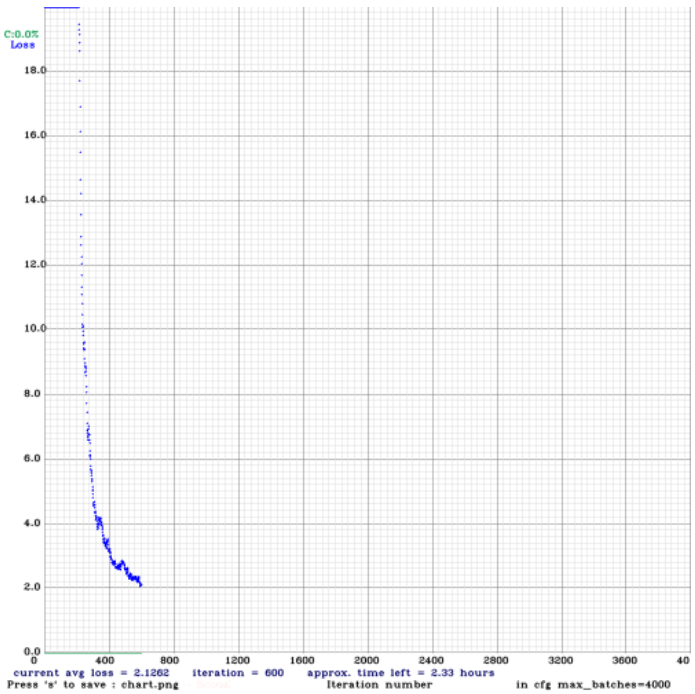
En este capítulo se presenta en profundidad el proceso de pruebas del sistema, desde la concepción de la propuesta hasta la evaluación de su eficiencia mediante las métricas arrojadas por el entrenamiento de YOLO. El objetivo es alcanzar los resultados y objetivos establecidos de manera exitosa, asegurando así el cumplimiento de estos.

3.1 Resultados del Entrenamiento de la red

En la Figura 40, se observa como la pérdida disminuye en cada etapa de entrenamiento. Después de 600 épocas, se logra alcanzar una pérdida de loss=2.162, con un tiempo aproximado de entrenamiento de time=2.33 hours, por lo tanto se puede concluir que, en un tiempo no muy prolongado se logró alcanzar una pérdida mínima con respecto al conjunto de validación.

Figura 41

Rendimiento del algoritmo de entrenamiento

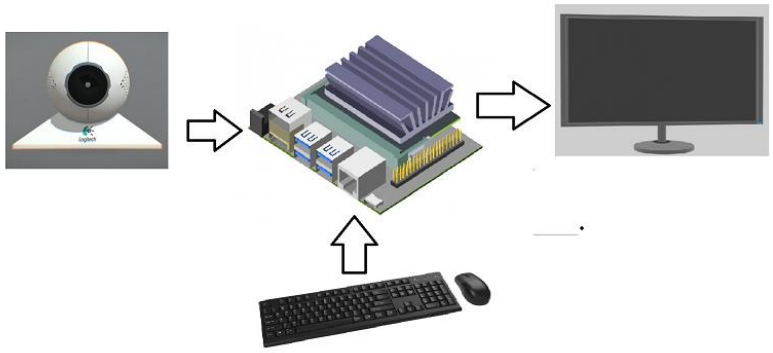


3.2 Diagrama de conexión basado en la arquitectura del sistema.

El sistema se conecta básicamente como un computador personal como se muestra en la Figura 41, puesto que la Jetson nano tiene puertos de comunicación USB, se conecta directamente la Camara, mouse, teclado y monitor para ejecutar el sistema.

Figura 42

Diagrama de conexión del prototipo



3.3 Ensamble del prototipo

El ensamble del sistema se realiza en primer lugar manufacturando la carcasa que consta de dos partes impresas en 3D que se sujetan con una ligera presión. En la Figura 42 se muestra que la placa Jetson ingresa fácilmente y coincide con los puestos de salida y entrada que se están en el diseño CAD.

Figura 43

Jetson Nano dentro de la carcasa inferior

Luego de encajar la placa en la carcasa inferior se procede a colocar la carcasa superior haciendo una ligera presión hasta que se logre sujetar como se muestra en la Figura 43.

Figura 44

Ensamble de la carcasa superior

Finalmente se alimenta el sistema y se procede a conectar la camara en el puerto cualquier puerto USB de la Jetson como se muestra en la Figura 44.

Figura 45

Conexión de Cámara y alimentación

3.4 Pruebas del sistema

El código del anexo 1 ejecuta el código del sistema dentro del entorno de programación de Spyder del Jetson Nano 2GB. En la Figura 45, se muestra que el trabajador está usando correctamente el casco de seguridad.

Figura 46

Prueba con casco de seguridad

El sistema detecta correctamente cuando un trabajador esta dejando de usar el casco de seguridad como se muestra en la Figura 46.

Figura 47

Prueba sin casco de seguridad

En la Figura 45 y 46 se observa la ejecución correcta del código y que detecta correctamente cuando se usan o no los cascos de seguridad

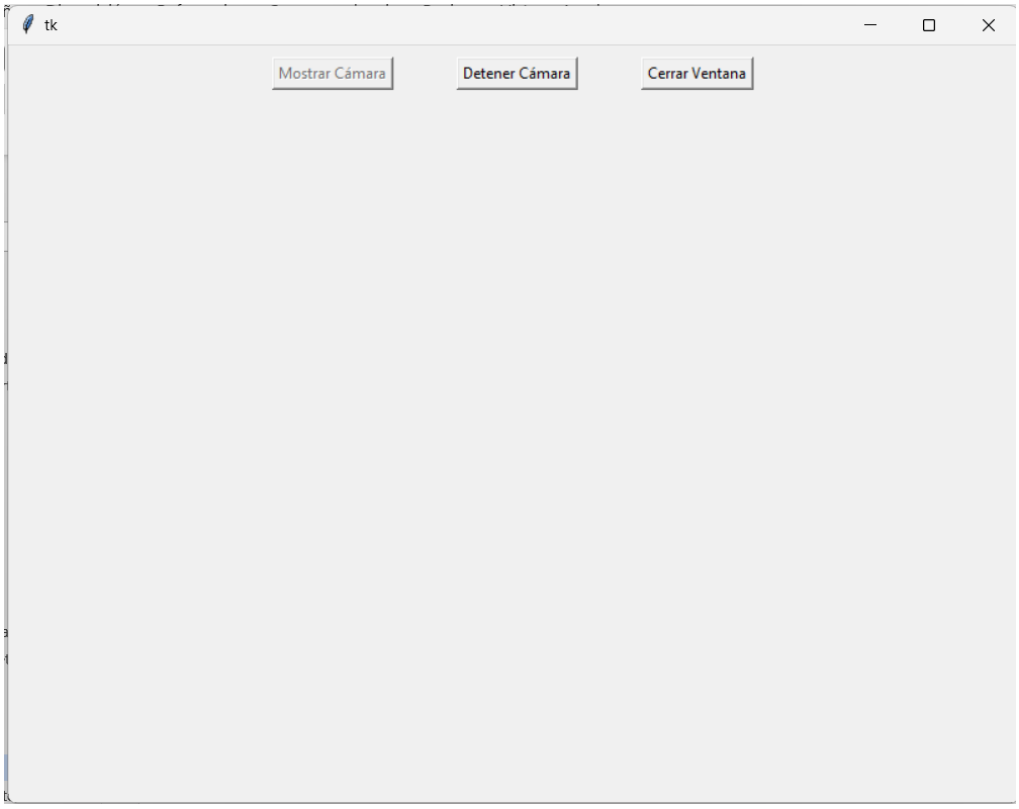
3.5 Interfaz de usuario

Para la obtención de la imagen de entrada para comprobar que el sistema funcione correctamente se utiliza una interfaz de usuario creada con la librería Tkinter que integra Python y que es compatible con el sistema embebido Jetson Nano 2GB.

En la Figura 47, se observa los botones de los que consta la interfaz gráfica y del espacio del label que contendrá la imagen de la cámara para realizar la detección del uso de los cascos de seguridad.

Figura 48

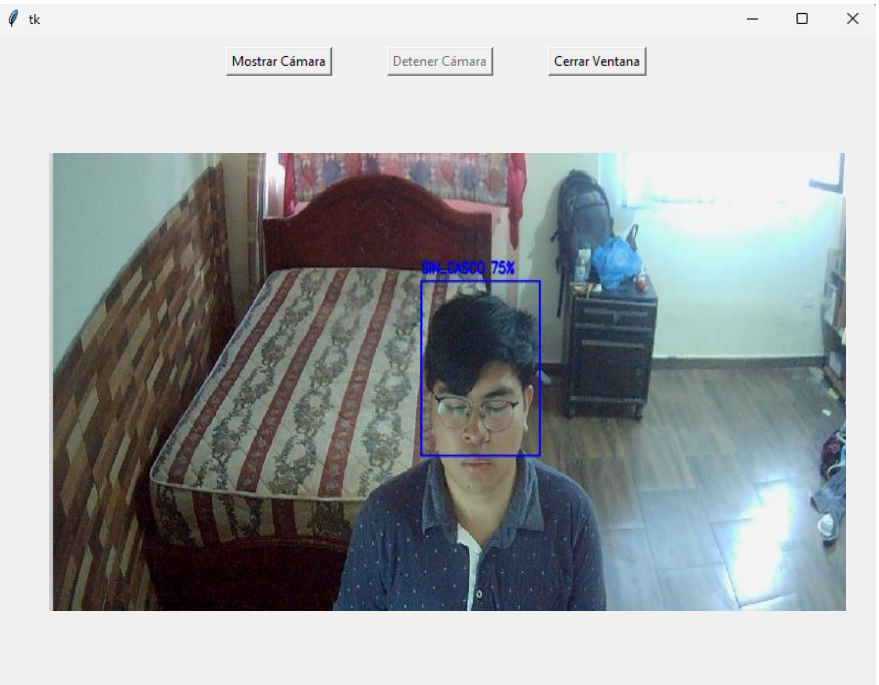
Interfaz Gráfica del sistema



En la Figura 48, se observa que al presionar el botón mostrar cámara se ejecuta el sistema mostrando el label de la cámara que proporciona la imagen de la cámara y los recuadros delimitadores del objeto que se busca detectar, en este caso los cascos de seguridad.

Figura 49

Ejecución del sistema desde la interfaz gráfica

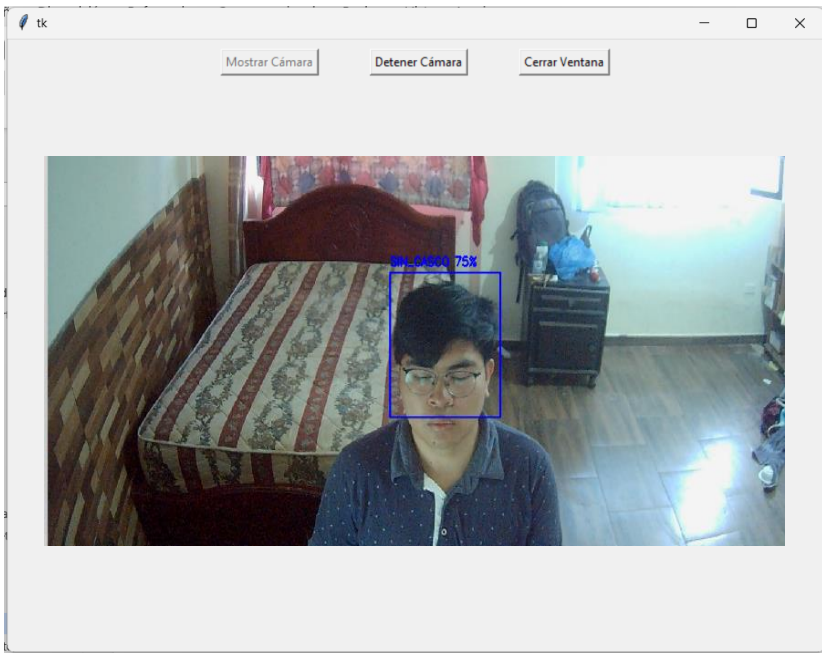


3.6 Pruebas del sistema en tiempo real con alarma sonora

El sistema funciona correctamente corriendo sobre la interfaz gráfica y emite una alerta sonora “Colócate el casco de seguridad”, por favor” cada vez que se ejecuta como se muestra en la Figura 49. En este tipo de sistemas las cámaras trabajan a 30 cuadros por segundo, esta cuestión hace que sea difícil implementar una lógica que permita mantener el sonido que detecta porque se crea un bucle infinito, para solucionar este problema se ha implementado un delay de frames que permita ejecutar el sonido una vez que detecta cada 7 fotogramas.

Figura 50

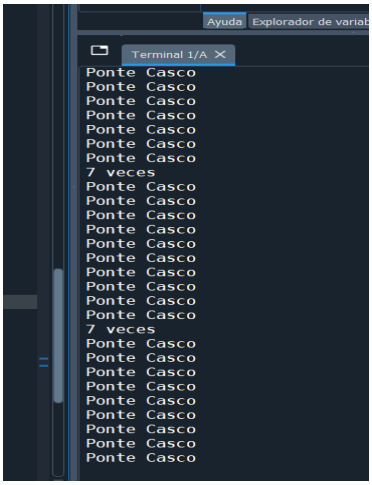
Prueba del sistema con sonido de alerta



Para comprobar que la ejecución de la sección de código de la señal sonora se ha implementado un contador que imprima el texto “sin casco”, con esto se puede saber si se está ejecutando el audio.mp3 del proyecto que indica al trabajador que debe colocarse el casco, esto se muestra en la Figura 50.

Figura 51

Impresión de texto indicando el sonido



CAPITULO IV

4. CONCLUSIONES Y RECOMENDACIONES

Durante el desarrollo de esta investigación, se realizaron una serie de actividades para lograr alcanzar los objetivos establecidos previamente. A continuación, se describe las conclusiones a las que se llega luego de haber cumplido los objetivos propuestos, así como las recomendaciones que se plantean en trabajos posteriores.

Conclusiones

- Se seleccionaron correctamente las especificaciones necesarias para la implemetaciòn de un sistema electrónico y de comunicación para detectar el uso de cascos de seguridad en los lugares de trabajo, cumpliendo con las necesidades y requisitos técnicos necesarios para llevar a cabo la implantación del sistema. Para garantizar el funcionamiento y eficiencia del prototipo, ha sido importante definir con precisión el uso tanto del sistema embebido como de la cámara.
- Se diseñó el sistema en base a los requisitos del primer objetivo, este proceso implicó la selección de componentes, la integración de los componentes por comunicación inalámbrica y alámbrica, además de la creación un modelo CAD que contenga el hardware del sistema. El resultado fue un diseño compacto que ayudo para la implementación exitosa del sistema.
- Se implementaron algoritmos de visión artificial para la detección del uso de cascos de seguridad usando la red neuronal YOLO. Se investigó diversas técnicas de procesamiento de imágenes y aprendizaje automático para lograr una detección precisa y confiable del uso de cascos de seguridad.

- Finalmente, las pruebas de funcionamiento ayudaron a la validación del funcionamiento del sistema en tiempo real. Se realizaron pruebas en entornos laborales simulados y reales para evaluar la precisión y la confiabilidad del sistema en la detección del uso de los cascos de seguridad. De esta manera el sistema logra cumplir con los objetivos establecidos y da una solución para mejorar la seguridad en entornos laborales que requieran el uso de cascos de seguridad.

Recomendaciones

- Se recomienda tomar fotografías con una cámara que tenga un ángulo de visión mayor a 90º para poder detectar los objetos en un área amplia sin tener problemas de enfoque en la imagen.
- Para seleccionar el modelo de red neuronal de Yolo a usar, se recomienda basarse en e hardware en el que va a ser implementado el programa de detección de objetos. Esto se hace para evitar problemas de rendimiento cuando la red se ejecute en tiempo real.
- El etiquetado de las imágenes debe hacerse con mucho cuidado, ya que cometer errores en este paso tan importante puede traer graves consecuencias en el modelo final de detección, con una o dos imágenes mal etiquetadas el programa deja de funcionar correctamente.

BIBLIOGRAFÍA

[1] N. d. R. Amagua, «Influencia de la implementación del Reglamento de Seguridad y Salud Ocupacional en la disminución de incidentes y accidentes laborales en los trabajadores de la Industria Comercial Plástica Mendieta Carrillo Cía. Ltda. - PLASTIMEC.,» Quito: UCE, Quito, 2014.

[2] IEES, Accidentes laborales, Quito: IEES, 2018.

[3] IESS, «Boletín Estadístico numero 24,» IESS, Quito, 2020.

[4] M. Massiris y F. Delrieux , «Detección de equipos de protección personal mediante red neuronal convolucional YOLO.,» *Actas de las XXXIX Jornadas de Automática,* pp. 5,7, 2020.

[5] A. Gonzáles y A. Pernía, Técnicas y Algoritmos Básicos de Visión Artificial, Anaya Multimedia, 2006.

[6] C. Cognex, «Introducción a la visión artificial,» 28 07 2016. [En línea]. Disponible en: https://www.cognex.com/.

[7] X. Wang, Artist, *A Safety Helmet and Protective Clothing Detection Method based on Improved-Yolo V 3.* [Art]. IEEE, 2020.

[8] F. Wilhelm, Artist, *Detecting motorcycle helmet use with deep learning.* [Art]. Elsevier, 2020.

[9] INEC, «Encuesta Nacional de Empleo, Desempleo y Subempleo,» 2020. [En línea]. Disponible en: https://www.ecuadorencifras.gob.ec/.

[10] C. E. I. y. P. Ministerio de Producción, «Boletín de cifras del sector productivo.,» Ministerio de Producción, Comercio Exterior, Inversión y Pesca., Quito, 2022.

[11] R. C. R. F. C. S. Jane de Lourdes Toro Toro, Artist, *Normativa en seguridad y salud ocupacional en el Ecuador*. [Art]. Universidad Regional Autónoma de Los Andes. , 2020.

[12] E. R. E. Javier, Artist, *El clima organizacional y su desempeño laboral de los trabajadores del Gad de Riobamba.* [Art]. Universidad Nacional de Chimborazo, 2022.

[13] R. G. B. Borja, Artist, *Recopilación de la normativa legal y jurisprudencia aplicable a enfermedades profesionales y accidentes de trabajo y elaboración de VADEMECUM de relcmos sobre riesgos de trabajo.*. [Art]. UNIVERSIDAD INTERNACIONAL SEK, 2014.

[14] M. d. Trabajo, Artist, *Ley Orgánica del Trabajo, los Trabajadores y las Trabajadoras.* [Art]. Ministerio del Trabajo, 2015.

[15] M. A. S. Flores, Artist, *Influencia de las características individuales y condiciones laborales en la gravedad de lesiones por Accidente de Trabajo en afiliados al Instituto Ecuatoriano de Seguridad Social (IESS) en la Provincia de Cañar, en el año 2014 y 2015.*. [Art]. UNIVERSIDAD DE CUENCA , 2018.

[16] S. U. Oviedo, Artist, *Propuesta de implementación de un sistema de gestion de seguridad industrial y salud en el trabajo para una empresa de distribución de televisión pagada en la ciudad de Quito.* [Art]. Universidad Católica del Ecuador, 2017.

[17] M. J. F. Rojo, Artist, *Manual básico de prevención de riesgos laborales: higiene industrial, seguridad y ergonomía.*. [Art]. Sociedad Asturiana de medicina y seguridad en el trabajo y Fundación de Médicos ASturias, 2000.

[18] E. M. J. Andrés, Artist, *Los equipos de protección personal y su incidencia en los riesgos laborales de los trabajadores del gobierno autónomo descentralizado del cantón Salcedo, provincia de Cotopaxi.*. [Art]. Universidad Técnica de Ambaro, 2016.

[19] M. J. D. Olmedo, Artist, *"Diagnóstico de riesgos laborales para la implementación de un plan de prevención de incidentes y accidentes en el Dept. Seguridad Ciudadana del Municipio de Quito – Zona Eloy Alfaro.* [Art]. Universidad Central del Ecuador, 2014.

[20] O. S. D. Manola, Artist, *Elaboración de un plan de seguridad e higiene industrial en la estación de Parahuacu de petroproducción distrito amazónico.*. [Art]. Escuela Superior Politécnica de Chimborazo, 2011.

[21] I. E. d. Normalización, Casco de seguridad para uso industrial, Quito: Instituto Ecuatoriano de Normalización .

[22] Barbieri, «Barbieri,» 17 05 2021. [En línea]. Disponible en: https://www.adbarbieri.com/blog/color-de-los-cascos-en-la-construccion.

[23] G. Calisma, «Grupo Calisma,» 28 07 2023. [En línea]. Disponible en: https://grupocasalima.com/es-ec/blog/colores-de-casco-que-debes-usar-segun-tu-cargo-o-profesion/.

[24] Henry Nelson Aguilera Vidal, Franklin Landerson Gallegos Ramírez, Anabell Martha Rea, «Casco inteligente de seguridad industrial para la prevención de accidentes y enfermedades ocupacionales.,» *InGenio,* vol. 4, nº 1, pp. 11-16, 2021.

[25] Serbusa, «Serbusa.net,» 28 07 2023. [En línea]. Disponible en: https://www.serbusa.net/2013/05/07/tecnologia-rfid/.

[26] V. Engineering, «vdl-eng.com,» 28 07 2019. [En línea]. Disponible en: https://www.vld-eng.com/blog/sistemas-de-vision-artificial/.

[27] L. A. López y G. Ramírez , «DISEÑO DE MÓDULO DE VISION PARA CELDA DE MANUFACTURA FLEXIBLE.,» 2014. [En línea]. Disponible en: https://ciateq.repositorioinstitucional.mx/jspui/bitstream/1020/218/1/Dise%C3%B1o%20de%20modulo%20de%20vision%20para%20celda.pdf.

[28] A. Duque, SISTEMA DE ATENCIÓN VISUAL PARA LA DETECCIÓN DE PUNTOS TOPOLÓGICOS DE REFERENCIA, Madrid : UNIVERSIDAD CARLOS III DE MADRID, 2009.

[29] R. Szeliski, Computer Vision: Algorithms and Applications, Springer, 2010.

[30] J. Graffigna, «unsj.edu.ar,» 31 07 2017. [En línea]. Disponible en: http://dea.unsj.edu.ar/imagenes/recursos/?C=D;O=D.

[31] J. Mejía, Procesamiento Digital de Imágenes, San Luís Potosí: UASLP, 2005.

[32] G. López , Artist, *Implementación de algoritmos de procesamiento de imágenes en FPGA.* [Art]. Instituto Politécnico Nacional, 2014.

[33] R. Cárdenes, «ulpgc.es,» 30 07 2023. [En línea]. Disponible en: https://www2.ulpgc.es/hege/almacen/download/38/38584/practica_ia_2.pdf.

[34] N. López, Artist, *Sistema automatizado de detección y clasificación mediante Deep Learninf de atributos dermatoscópicos en lesiones de la piel para diagnóstico de Melanoma.* [Art]. Escuela Superior de Ingeniería y Telecomunicaciones, 2021.

[35] E. García, Artist, *Introducción a las redes neuronales deconvolución. Aplicación a la visión por ordenador.* [Art]. Universidad Zaragoza, 2019.

[36] C. Bonilla, Artist, *REDES CONVOLUCIONALES.* [Art]. 2020.

[37] P. Erazo y G. Navarrete, Artists, *Detección de pistas aéreas ilegales en imágenes digitales empleando técnicas de inteligencia artificial.* [Art]. Escuela de Postgrados, Fuerza Aérea Colombiana, 2022.

[38] J. Redmon, S. Divvala, R. Girshick y A. Farhadi, Artists, *You Only Look Once: Unified, Real-Time Object Detection.* [Art]. University of Washington, 2016.

[39] R. Singh, «Edge AI: A survey,» *ScienceDirect,* p. 72, 2023.

[40] S. Hymel, «https://www.digikey.com,» 26 08 2023. [En línea]. Disponible en: https://www.digikey.com/en/maker/projects/what-is-edge-ai-machine-learning-iot/4f655838138941138aaad62c170827af.

[41] 4. VENCO, «Vencoel.com,» 16 02 2023. [En línea]. Disponible en: https://www.vencoel.com/que-es-edge-ai-y-como-esta-revolucionando-la-industria/.

[42] Rafael Aracil Santonja, Alberto Sanfeliu Cortés, Visión Artificial y Robótica, Alianza Editorial, 2002.

[43] S. Luchetti, «tribalyte.eu,» 29 09 2021. [En línea]. Disponible en: https://tech.tribalyte.eu/blog-sistema-embebido-caracteristicas#El_sistema_embebido_Que_es_exactamente. [Último acceso: 01 09 2023].

[44] D. Roca, Artist, *Diseño de un sistema de actualización de firmware para.* [Art]. Instituto Universitario Aeronáutico (I.U.A.), 2017.

[45] Arduino, «arduino.cc,» 01 09 2023. [En línea]. Disponible en: https://www.arduino.cc. [Último acceso: 01 09 2023].

[46] Beagleboard, «beagleboard.org,» 01 09 2023. [En línea]. Disponible en: https://www.beagleboard.org/boards/beaglebone-black. [Último acceso: 01 09 2023].

[47] Raspberry Pi, «raspberrypi.org,» 01 09 2023. [En línea]. Disponible en: https://www.raspberrypi.org/help/what-%20is-a-raspberry-pi/. [Último acceso: 01 09 2023].

[48] Nvidia, «nvidia.com,» 01 09 2023. [En línea]. Disponible en: https://developer.nvidia.com/embedded/jetson-nano-developer-kit#:~:text=NVIDIA®%20Jetson%20Nano™,as%20little%20as%205%20watts.. [Último acceso: 01 09 2023].

[49] D. V. Jesús Briales, Introducción a la Visión por Computador: Guía Práctica para Resolver Problemas, Ediciones ENI, 2014.

[50] M. Á. P. Orfila, Visión Artificial: Métodos y Aplicaciones, Ediciones Díaz de Santos, 2017.

[51] I. Rondón, «eiposgrados.com,» 31 07 2023. [En línea]. Disponible en: https://eiposgrados.com/blog-python/que-es-anaconda/.

[52] D. Arevalo , «ubunlog.com,» 31 07 2023. [En línea]. Disponible en: https://ubunlog.com/spyder-entorno-desarrollo-python/.

ANEXOS

Anexo 1: Código de ordenamiento de imágenes

```python
import cv2 as cv
import numpy as np
#import matplotlib.pyplot as plt
import glob

paths =glob.glob("Imagenes_tomadas/*.jpg")


count = 0
for path in paths:
    img = cv.imread(path)
    #new_img = cv.resize(img, (100,40))
    cv.imwrite("Dataset/img"+str(count)+".jpg",img)
    count+=1
```

Anexo 2: Código de ejecución del sistema

```python
import cv2
import numpy as np

# ---------------- READ DNN MODEL ----------------
# Model configuration
config = "cfg/Cascos.cfg"
# Weights
weights = "cfg/Cascos_4000.weights"
# Labels
LABELS = open("cfg/Cascos.names").read().split("\n")
#print(LABELS, len(LABELS))
colors = np.random.randint(0, 255, size=(len(LABELS), 3), dtype="uint8")
#print("colors.shape:", colors.shape)

# Load model
net = cv2.dnn.readNetFromDarknet(config, weights)

# ---------------- READ THE IMAGE AND PREPROCESSING ----------------
image = cv2.imread("Images/img0.jpg")
height, width, _ = image.shape

# Create a blob
blob = cv2.dnn.blobFromImage(image, 1 / 255.0, (416, 416),
                             swapRB=True, crop=False)
#print("blob.shape:", blob.shape)

# ---------------- DETECTIONS AND PREDICTIONS ----------------
ln = net.getLayerNames()
#print("ln:", ln)

# ln = [ln[i[0] - 1] for i in net.getUnconnectedOutLayers()]
ln = [ln[i - 1] for i in net.getUnconnectedOutLayers()]
#print("ln:", ln)
```

```
net.setInput(blob)
outputs = net.forward(ln)
#print("outputs:", outputs)

boxes = []
confidences = []
classIDs = []

for output in outputs:
     for detection in output:
          #print("detection:", detection)
          scores = detection[5:]
          classID = np.argmax(scores)
          confidence = scores[classID]

          if confidence > 0.5:
               #print("detection:", detection)
               #print("classID:", classID)
               box = detection[:4] * np.array([width, height, width, height])
               #print("box:", box)
               (x_center, y_center, w, h) = box.astype("int")
               #print((x_center, y_center, w, h))
               x = int(x_center - (w / 2))
               y = int(y_center - (h / 2))
               #cv2.rectangle(image, (x, y), (x + w, y + h), (0, 255, 0), 2)

               boxes.append([x, y, w, h])
               confidences.append(float(confidence))
               classIDs.append(classID)

idx = cv2.dnn.NMSBoxes(boxes, confidences, 0.5, 0.5)
print("idx:", idx)
```

```
if len(idx) > 0:
     for i in idx:
          (x, y) = (boxes[i][0], boxes[i][1])
          (w, h) = (boxes[i][2], boxes[i][3])

          color = colors[classIDs[i]].tolist()
          text = "{}: {:.3f}".format(LABELS[classIDs[i]], confidences[i])
          cv2.rectangle(image, (x, y), (x + w, y + h), color, 2)
          cv2.putText(image, text, (x, y - 5), cv2.FONT_HERSHEY_SIMPLEX,
                      0.5, color, 2)

cv2.imshow("Image", image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```