ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

CARRERA TELECOMUNICACIONES

"IMPLEMENTACIÓN DE UN PROTOTIPO DE EMISIÓN DE ALERTAS EN ZONAS DE PELIGRO PARA EL CUIDADO DE NIÑOS DE 1 A 5 AÑOS MEDIANTE VISIÓN ARTIFICIAL."

Trabajo de Titulación

Tipo: Proyecto Técnico

Presentado para optar al grado académico de:

INGENIERO EN TELECOMUNICACIONES

AUTOR:

JEFFERSON TOBÍAS GARCÍA SANUNGA

Riobamba – Ecuador

2023

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

CARRERA INGENIERÍA EN TELECOMUNICACIONES

"IMPLEMENTACIÓN DE UN PROTOTIPO DE EMISIÓN DE ALERTAS EN ZONAS DE PELIGRO PARA EL CUIDADO DE NIÑOS DE 1 A 5 AÑOS MEDIANTE VISIÓN ARTIFICIAL."

Trabajo de Titulación

Tipo: Proyecto Técnico

Presentado para optar al grado académico de:

INGENIERO EN TELECOMUNICACIONES

AUTOR: JEFFERSON TOBÍAS GARCÍA SANUNGA

DIRECTOR: ING. JEFFERSON ALEXANDER RIBADENEIRA RAMIREZ, Ph.D.

Riobamba – Ecuador

2023

Yo, Jefferson Tobías García Sanunga, declaro que el presente trabajo de integración curricular es de mi autoría y los resultados del mismo son auténticos. Los textos en el documento que provienen de otras fuentes están debidamente citados y referenciados.

Como autor asumo la responsabilidad legal y académica de los contenidos de este trabajo de integración curricular; el patrimonio intelectual pertenece a la Escuela Superior Politécnica de Chimborazo.

Riobamba, 7 de marzo de 2023

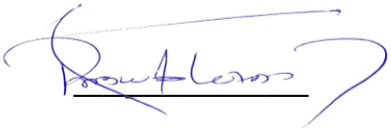
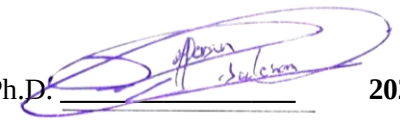
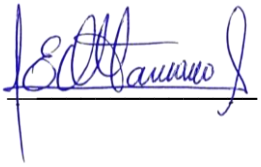
Jefferson Tobías García Sanunga
020232566-8

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

CARRERA TELECOMUNICACIONES

El tribunal de Trabajo de Titulación certifica que: El Trabajo de Titulación; Tipo: Proyecto Técnico; **IMPLEMENTACIÓN DE UN PROTOTIPO DE EMISIÓN DE ALERTAS EN ZONAS DE PELIGRO PARA EL CUIDADO DE NIÑOS DE 1 A 5 AÑOS MEDIANTE VISIÓN ARTIFICIAL**, realizado por el señor: **JEFFERSON TOBÍAS GARCÍA SANUNGA**, ha sido minuciosamente revisado por los Miembros del Tribunal del Trabajo de Integración Curricular, el mismo que cumple con los requisitos científicos, técnicos, legales, en tal virtud el Tribunal Autoriza su presentación.

	FIRMA	**FECHA**
Ing. Pablo Lozada **PRESIDENTE DEL TRIBUNAL**		**2023-04-20**
Ing. Jefferson Alexander Ribadeneira Ramírez, Ph.D. **DIRECTOR DE TRABAJO DE TITULACIÓN**		**2023-04-20**
Ing. Edwin Vinicio Altamirano, MSc. **ASESOR DE TRABAJO DE TITULACIÓN**		**2023-04-20**

DEDICATORIA

Dedico con todo mi corazón a mis padres, hermanos, hermanas y a mi prometida, quienes fueron los pilares fundamentales para formar mi moral, y por ser el motor principal por el que siempre encontraba una razón para continuar.

Tobías

AGRADECIMIENTO

De lo más profundo de mi corazón quiero agradecer a mis padres, quienes siempre fueron el empuje y las ganas que siempre me faltaron. A mis hermanos y hermanas por estar pendientes de mi en cada momento y nunca dejarme solo. A mi prometida que con su comprensión y carisma siempre vio lo mejor en mí y día a día me ayudaba a demostrarlo y nunca me dejó desfallecer.

Tobías

ÍNDICE DE CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE ILUSTRACIONES

ÍNDICE DE ANEXOS

RESUMEN

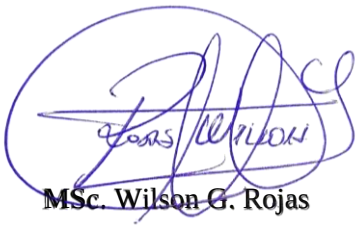
El presente trabajo de titulación engloba todos los criterios de visión artificial optando por una tecnología que permita realizar este proceso en tiempo real dentro de áreas determinadas como zonas de peligro, en cada zona los objetos que requieren ser detectados son diferentes y por esta razón se seleccionó una tecnología que pueda ser entrenada de forma personalizada. Se trabajó con la tecnología Yolo v5 para la detección de objetos debido a su capacidad de detectar en tiempo real y a gran velocidad. Por su lado pytorch en el entrenamiento genera un aprendizaje profundo en la última red de neuronas. En base al entrenamiento personalizado para cada zona de peligro se obtuvo una matriz de confusión que muestra el porcentaje de falsos positivos cuando un objeto fue detectado sin estar presente en el escenario de validación. Por lo tanto el porcentaje de error más alto en la primera zona de peligro le corresponde al objeto *tenedor* con una confusión de 47% y en la segunda zona de peligro a la clase *Hijo(a)* con un valor de 42% de confusión. En los escenarios de prueba el sistema mostró buenos resultados detectando cada objeto de forma rápida y precisa, gracias a esto se pudo determinar la posición exacta de cada elemento detectado para poder calcular la distancia entre el objeto de interés y el niño. Si la distancia entre la clase Hijo(a) y el objeto de peligro es menor a 300 píxeles correspondiente a un metro de distancia, el tutor es notificado por dos diferentes métodos, una alerta emitida por mensaje mediante la plataforma de Telegram con un retraso de 1 segundo y otra alerta de forma sonora con un retraso de 2 segundos.

Palabras clave: <VISIÓN ARTIFICIAL>, <ZONAS DE PELIGRO>, <EMISIÓN DE ALERTAS>, <ENTRENAMIENTO PERSONALIZADO>.

SUMMARY

The present degree work encompasses all the artificial vision criteria, opting for a technology that allows this process to be carried out in real time within certain areas such as danger zones, in each zone the objects that need to be detected are different and for this reason a technology that can be trained in a personalized way was selected. Yolo v5 technology was used for object detection due to its ability to detect in real time and at high speed. For its part, pytorch in training generates deep learning in the last network of neurons. Based on the personalized training for each danger zone, a confusion matrix was obtained that shows the percentage of false positives when an object was detected without being present in the validation scenario. Therefore, the highest error percentage in the first danger zone corresponds to the fork object with a confusion of 47% and in the second danger zone to the Child class with a value of 42% confusion. In the test scenarios the system showed good results detecting each object quickly and accurately, thanks to this it was possible to determine the exact position of each detected element in order to calculate the distance between the object of interest and the child. If the distance between the Child class and the object of danger is less than 300 pixels corresponding to one meter away, the tutor is notified by two different methods, an alert issued by message through the Telegram platform with a delay of 1 second and another audible alert with a 2 second delay.

Keywords: <ARTIFICIAL VISION>, <ZONA ZONES>, <ALERT ISSUANCE>, <PERSONALIZED TRAINING>.



MSc. Wilson G. Rojas

NOMBRE Y FIRMA PROFESOR

C.I 0602361842

INTRODUCCIÓN

Sin duda, el hogar es el ambiente en el que mayor tiempo pasamos en el lapso de nuestras vidas las pequeñas cosas que pasan desapercibidas pueden resultar en términos de probabilidad las causantes de daños en la integridad física de cada uno de los miembros, y si bien, todos los integrantes de la familia pueden resultar heridos por cortes de cuchillos, vidrios de vasos rotos, tijeras, tenedores, estiletes, planchas, etc. Son los niños que, por efecto de aprendizaje a temprana edad y curiosidad como parte del crecimiento, son quienes están más expuestos a recibir cualquier tipo de trauma por cada uno de los utensilios anteriormente expuestos.

De contar con un ambiente plenamente seguro se evitarían muchos desastres en zonas que representan un peligro para los niños, por lo cual en búsqueda de prever y anticipar cualquier accidente, tanto los padres como los tutores y cuidadores de los más pequeños se ven en la necesidad de establecer sistemas de monitoreo por cámaras de vídeo para cuidar la integridad de los pequeños, pero las ocupaciones diarias siempre dan espacio a descuidos. Es imprescindible entonces contar con un sistema de alerta y monitoreo en tiempo real, que notifique al cuidador de un peligro inminente.

Por contraparte, la visión artificial ha evolucionado de tal manera que puede identificar de manera específica y muy precisa varios objetos específicos, mediante una cámara que actúa como el ojo del ordenador y siendo su retina los sensores que reaccionan a diferentes tonos de luminosidad. Como parte de esta disciplina, se explora la detección de objetos, que es una técnica basada en visión por computadora para identificar y localizar objetos dentro de una imagen o vídeo.

Cumpliendo con lo anterior, Yolo es una tecnología que se basa en la detección de objetos y cumple con el concepto de visión artificial, tecnología que al suministrarle la dataset correspondiente personalizado en conjunto con librerías y paqueterías puede detectar objetos de interés. A esto se le suma una programación específica en python para que la visión artificial pueda detectar la proximidad entre un niño y un objeto peligroso (vasos de vidrio, cuchillos, tijeras, tenedores, etc.) para la integridad de los niños y alerte mediante sonido y mensajes al cuidador.

CAPÍTULO I

1. DIAGNÓSTICO DEL PROBLEMA

1.1. Antecedentes

El desarrollo de las tecnologías que requieren de inteligencia artificial para poder procesar imágenes y realizar una correcta detección de objetos dependerán siempre de los sistemas computarizados. Por esto a este proceso se le conoce también visión por computador, qué, en realidad no son más que sistemas que simulan el proceso del sentido de la visión, mediante el análisis y la interpretación de los objetos que integran una escena en una imagen adquirida. El crecimiento continuo de aplicaciones relacionadas a la visión artificial se debe al desarrollo de los lenguajes de programación y los sistemas operativos. Es común encontrar diseños que realicen detección y seguimiento de objetos, análisis de imágenes, autentificación, reconocimiento de rostros y huellas dactilares, control de calidad de productos, etc.

Como evidencia de procesos anteriores (Oliva, 2018) al hacer uso de las redes neuronales se obtiene una detección de objetos extraídos de imágenes, estas redes fueron entrenadas y alimentadas con una cantidad de datos y registros personalizados, permitiéndoles así clasificar determinados parámetros de forma ordenada, el desarrollo de la herramienta permite al usuario la creación y manipulación de un conjunto de datos de forma gráfica.

Otros estudios, como (Gonzales, 2017) en la ciudad de México, ponen a prueba el sistema con un banco de imágenes almacenadas previamente, se realiza una comparación entre las imágenes almacenadas y una imagen de entrada, mostrando buenos resultados al momento de clasificar y reconocer objetos. Así facilita el recordar la gran cantidad de nombres que muchos objetos pueden tomar en especial si se tratan de herramientas. Por tanto, el objetivo de esta aplicación es reconocer estos objetos a partir de una imagen brindando información como nombres, existencia, precio, ubicación, etc.

Por otro lado, en La Universidad Técnica de Ambato por (Chicaiza, 2017) su visión sobre inteligencia artificial es un poco más guida al cuidado de la integridad humana proporcionando más información al momento de conducir un vehículo gracias a que mediante alertas audibles proporciona al conductor información ante la presencia de una señal de tránsito cercana, siendo capaz incluso de detallar la distancia a la que se encuentra el vehículo con respecto a la señal de tráfico y el respectivo nombre de la señal.

1.2. Planteamiento Del Problema.

En base a los requerimientos actuales y debido al gran avance de la tecnología, la factibilidad de poder detectar en tiempo real no solo un objeto, sino varios a le vez a través de cámaras es un hecho, pero implica cierta dificultad el poder encontrar una tecnología que abarque un ambiente cotidiano como una ayuda en el cuidado de niños y así poder evitar cualquier riesgo en su propio domicilio.

Por esto, se buscan tecnologías de detección de objetos en tiempo real que administre un dataset personalizado y aprenda a reconocer objetos de interés para reducir el costo computacional y obtener como resultado final un modelo apegado a necesidades específicas. Por tanto, el presente proyecto ha desarrollado un prototipo de detección de objetos en tiempo real mediante visión artificial para el cuidado de niños de 1 a 5 años en zonas de peligro que sea capaz de emitir alertas en tiempo real.

1.2.1. Formulación Del Problema.

¿Cómo implementar un prototipo de emisión de alertas en zonas de peligro para el cuidado de niños de 1 a 5 años mediante visión artificial?

1.2.2. Sistematización Del Problema

- ¿Cómo identificar si Pytorch es mejor que Yolo o viceversa?
- ¿Qué objetos representan un peligro para los niños de 1 a 5 años para poder clasificarlos como zona de peligro?
- ¿Cómo entrenar una red neuronal y convolucional para poder diseñar el prototipo?
- ¿Cómo validar el prototipo para identificar los resultados obtenidos?

1.3. Justificación.

1.3.1. Justificación Teórica.

La visión por computador está obteniendo avances importantes en los últimos años. La puesta en escena de las redes neuronales profundas y más concretamente de las redes neuronales convolucionales han dado paso a este salto de nivel. El reconocimiento facial, de señales de

tráfico o los coches autónomos son algunos de los ejemplos de las aplicaciones que se están consiguiendo.

Hay que tener en cuenta que para la clasificación de imágenes existen métodos de extracción de características para luego hacer uso del algoritmo de clasificación para las características extraídas. Sin embargo, antes de aplicar estos pasos se realiza un preproceso a la imagen para normalizar los efectos de contraste y brillo eliminando el ruido (Oliva, 2018, p.5).

Actualmente los sistemas de visión por computadora hacen uso de diferentes exploradores de imágenes en conjunto con cámaras de estado sólido, que juegan un papel muy importante para todos los procesos de captación de imágenes. Aunque, por contraparte, la selección del dispositivo de captación está estrechamente relacionado a la aplicación. Por ejemplo, si la aplicación consiste en la lectura de películas fotográficas, el captor más apropiado será un densitómetro. Si se quiere realizar a través de imágenes el estudio ambiental de una zona determinada un sensor adecuado podría ser una cámara en cualquiera de las siguientes versiones; fotográfica, de televisión o de digital. En esta aplicación el explorador o escáner queda excluido. Si en otra situación lo que se desea es realizar la restauración de documentos antiguos un explorador con una alta resolución sería el captor más apropiado (Gonzales, 2017, p.21).

1.3.2. Justificación Práctica.

El querer contar con un ambiente completamente seguro y que brinde una total respaldo a la integridad de niños de 1 a 5 años, se ve afectado por la falta disponibilidad de tiempo, porque dadas las ocupaciones diarias y profesionales de los padres o cuidadores no se presta completa atención al cuidado de los niños, su proximidad contra ventanas, objetos cortantes, escaleras y demás objetos pueden atentar contra su bienestar e integridad física, Por tanto es necesario , es necesario una asistencia de visión artificial como ayuda al cuidado paterno.

Usar visión artificial para hacer la detección de objetos se ha propuesto ya que gracias a esto se puede realizar en tiempo real y con muy buenos resultados y ayuda que brindan cada una de las paqueterías existentes en Python y la capacidad de hacer uso de redes neuronales para el entrenamiento de los objetos a identificar, es de gran ayuda para medir la proximidad entre objetos, personas a objetos, personas a otras personas. Al obtener cada una de las coordenadas de medición será fácil identificar su posición en el medio estudiado, para así finalmente enviar una alerta con estas coordenadas al usuario final.

Para llevar a cabo el sistema, se entrenarán las redes neuronales con cada uno de los objetos que deben ser detectados como indica el modelo de la ilustración 1-1, estos objetos son parte de un modelo computacional basado en las neuronas del sistema biológico, con la finalidad de reproducir de forma aproximada, el comportamiento de estas (IBM, 2017). Funciona con un elevado número de unidades interconectadas nombradas neuronas, también será de mucha ayuda librerías de Python como opencv y tensorflow que se en cargan del reconocimiento de cada imagen solicitada (OpenCV, 2018).

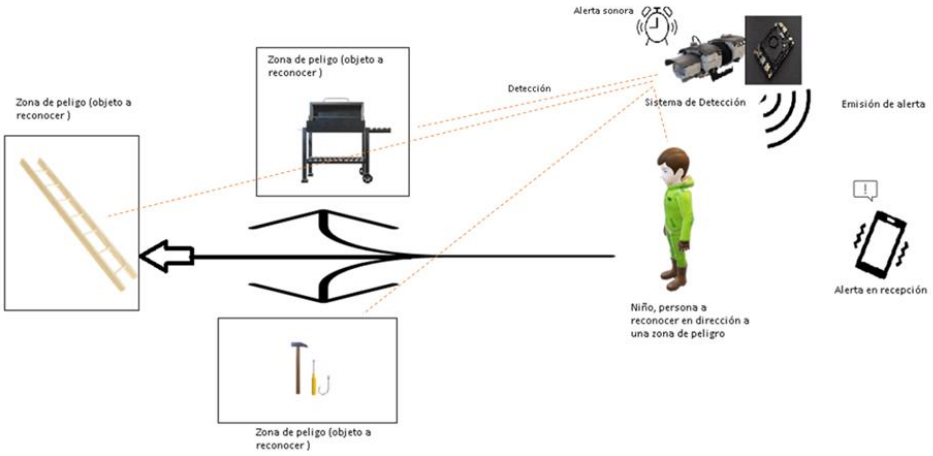


Ilustración 1-1: Esquema gráfico del proceso de detección en tiempo real

Realizado por: García, Jefferson, 2022

1.4. Objetivos.

1.4.1. Objetivo General.

Implementar un prototipo de emisión de alertas en zonas de peligro para el cuidado de niños de 1 a 5 años mediante visión artificial.

1.4.2. Objetivos Específicos.

- Seleccionar la tecnología de detección de objetos a utilizar entre Yolo y Pytorch
- Determinar las zonas de peligro para niños de 1 a 5 años dentro del domicilio
- Diseñar e implementar el prototipo con el entrenamiento respectivo para la de detección de objetos y emisión de alertas al cuidador en tiempo real.
- Validar la efectividad del prototipo para detectar el ingreso del niño a una zona de peligro.

CAPITULO II

2. MARCO TEÓRICO.

En este capítulo se hace referencia a los conceptos teóricos que servirán de fundamento para entender la investigación propuesta, sus beneficios y campos de aplicación. También se contempla la conceptualización general del hardware y software que se empleó en el desarrollo del proyecto.

2.1. Antecedentes De La Investigación.

En Madrid se muestra el trabajo de "*Detección de víctimas mediante redes neuronales e imágenes térmicas*", como desarrollar un sistema para detectar personas que pueden resultar víctimas en un entorno de desastre a partir de diferentes materiales, tales como escombros, fundamentando su investigación en tecnologías como YOLOv3, Faster R-CNN y SSD, que son herramientas muy útiles de modelos pre entrenados enfocados en la detección de víctimas, como enfoque principal, destacan la alta precisión, rapidez y baja pérdida de YOLOv3. No obstante, no se puede obviar que el proyecto basa su programación en python y sus múltiples librerías de Deep Learning y Visión Artificial, como Tensorflow, Pytorch y OpenCV. El reto latente aquí es que las redes neuronales siempre establecen su capacidad de identificar los objetos deseados en una cantidad enorme de datos en forma de imágenes, para ello, genera su propia dataset haciendo uso de una cámara infrarroja sobre varios escenarios, simulando victimas en cada uno de ellos, tomando como ejemplo cada parte del cuerpo humano y usando diferentes etiquetas correspondientes a los brazos, cabeza, piernas, torso, y a la persona misma mediante el software LabelImg (Prieto, 2021).

Mientras que en la universidad Técnica de Machala de Ecuador, (Córdova, 2020) en su proyecto "*Implementación de un sistema de reconocimiento de distanciamiento social como medida preventiva para COVID 19 usando Deep Learning*", mediante YOLOv3 con un modelo de detección previamente entrenado con el conjunto de imágenes COCO debido a su gran rendimiento y predicciones bastante precisas, se logra evaluar el rendimiento del sistema para extraer la distancia correcta entre cada individuo sobre imágenes y videos al variar la cantidad de pixeles entre cada recuadro que sirve como identificador de la clase persona. Al lograr encerrar la clase de interés dentro de un recuadro, lo que resta es identificar dos coordenadas dentro del recuadro, coordenadas que estan dadas en valores de pixeles indicando la posición

del objeto reconocido, al lograr identificar la clase y sus coordenadas, solo resta identificar la cantidad de pixeles exactos entre cada clase para finalmente establecer una distancia prudente.

2.2. Referencias Teóricas.

Se utilizan para mostrar el estado actual del conocimiento en el tema referente a la investigación realizada. Aquí se retribuye el crédito a los investigadores y teóricos que han trabajado en el tema antes y cuyos trabajos han sido utilizados en la investigación. En resumen, las referencias teóricas presentadas a continuación son esenciales para mostrar la relevancia y validez de la investigación realizada en la tesis.

2.2.1. Accidentes En Casa.

La Organización Mundial de la Salud (OMS) define accidente como un acontecimiento no premeditado que produce daño o lesión reconocible o visible corporal o mental. La Real Academia de la Lengua Española lo define como un suceso eventual, del que involuntariamente resulta un daño. El termino accidente está quedando en desuso ya que lleva implícito el concepto de que es algo irreversible que sucede al azar por causa del destino o suerte, hoy se utiliza más el termino de lesiones, intoxicaciones, traumatismos, quemaduras. (Carmilema, 2017, p.15).

El número de accidentes de diversa índole que se produce en el mundo es cada vez mayor. Sin embargo, los que ocurren en el hogar en niños pequeños pasan inadvertidos, pese a que constituyen un problema importante en el que intervienen factores culturales, sociales y enconómicos de la familia. A esto se agregan los rasgos propios del niño pequeño, quien tiene poca experiencia o juicio para limitar su afán exploratorio y reconocer el peligro que le rodea, así como el abandono y la falta de medidas de protección dentro del hogar (Minchola de Pérez, 1984, p. 414-416)

2.2.2. Tipos De Accidentes En El Hogar Y Factores.

Definitivamente las causas de accidentes son descuido de los padres o tutores encargados de los menores y en esto interfieren factores de riesgo que se encuentran presentes de manera imperceptible. La mayoría de las personas consideran a sus hogares como el sitio más seguro. Sin embargo, es el lugar donde se presentan el mayor número de casos de caídas en su propia altura, quemaduras, heridas, intoxicaciones, envenenamiento, asfixia, mordeduras, entre otras,

que son considerados los responsables de las principales atenciones médicas dentro del nivel de salud primario y secundario (Fiorentino, Molise & Stach, 2014, p.14-16).

Muchos factores que se encuentran en los hogares son aparentemente inofensivos a simple vista, La impericia de los padres hace que expongan a los menores a riesgos domiciliarios, entre ellos, la falta de cercas o puertas que impiden el acceso a la zona de cocina, así como sus utensilios, entre ellos cuchillos, tenedores, fundas plásticas donde pueden ser víctimas de quemaduras, asfixia, heridas. La presencia de objetos rotos o con filos cortantes, adornos frágiles, estiletes, tijeras, agujas, entre otros, que generalmente se encuentran al alcance de los niños, podrían desencadenar heridas (Ponce, 2017, p. 14).

2.2.3. Rango De Edad Que Necesita Más Prevención.

Los resultados confirman que los niños en su primer año de vida están relativamente seguros en manos de las personas que los cuidan. (Minchola de Pérez, 1984, p. 419). A medida que el menor crece, aumenta sus habilidades, su curiosidad y ganas de explorar, lo cual aumenta el riesgo de accidentes (Justo, 2018, p. 11).

Innumerables investigaciones coinciden que la frecuencia de los accidentes domésticos se presenta en infantes menores de cinco años, principalmente los que son muy extrovertidos, vivarachos e intranquilos, haciéndolos más vulnerables a los traumatismos que generalmente son golpes de cabeza, cuyas consecuencias dependerán del su entorno en el que se desenvuelven, así como su desarrollo psicomotor. Los riesgos de accidentes como caídas y heridas se ven reflejados en los más pequeños (Domingos, Alves & López, 2011, p. 12).

2.2.4. Visión Artificial.

Al hablar hipotéticamente de otorgarle ojos a una máquina, estamos englobando todos los procesos y elementos que se apegan al concepto de visión artificial, poniendo en evidencia que esta disciplina traduce y deduce de forma automática estructuras y propiedades de un mundo tridimensional y dinámico basándose en varias imágenes previamente tomadas y tratadas bidimensionalmente del mismo mundo. La visión, tanto para el hombre como para la máquina consta de dos etapas, captar una imagen y procesarla con la finalidad de entenderla, pero, si bien es cierto que existe un nivel de complejidad en el ojo humano que hace que se realice todo este proceso de forma natural. Para los ordenadores se enfoca de la siguiente forma, quien actúa como ojo para el ordenador es la cámara, y su retina un sensor que es sensible a la intensidad

luminosa, así que a la visión por computadora lo que le resta es tener la capacidad de entender las imágenes , diferenciar lo múltiples objetos puestos en escena, extraer la información pertinente y resolver las necesidades que se deseen satisfacer, como explicación de esto se expresa en la ilustración 1-2 un proceso por etapas de lo que realiza la visión artificial (Gonzáles, Martínez & Alba, 2006, p. 11-16.)

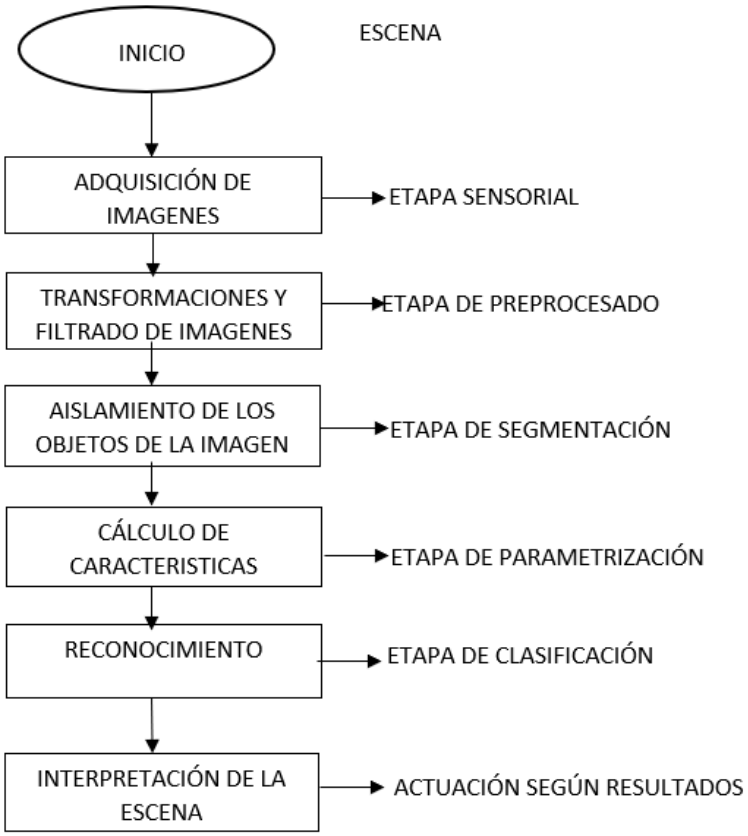


Ilustración 1-2: Diagrama de bloques de las etapas de un sistema de visión artificial

Fuente: (Gonzáles, Martínez & Alba, 2006, p. 16.)

2.2.5. Sistema de Alarmas

Para (Cuzco & Layana, 2012, p 15) el objetivo principal de un sistema de alarmas se basa en la funcionalidad práctica de detectar cualquier situación de riesgo que esté siendo presentada en cualquier ambiente. Por ello, un sistema de alarmas no significa únicamente la detección de un problema determinado, sino también, un conjunto de procesos que puedan responder de forma inmediata poniendo en aviso a las personas encargadas, ya sea al administrador de todo el sistema o cualquier empresa dedicada a solucionar este tipo de inconvenientes. Estas alertas ponen en manifiesto cualquier eventualidad anormal dentro del ambiente de interés que se desee

monitorear, siendo capaz de ser representadas por cualquier tipo de señal, sea sonido, imagen o texto como se aprecia en la ilustración 2-2.

Ilustración 2-2: Sistema de Alarma

Fuente: (Cuzco & Layana, 2012, p 15)

2.2.6. Python.

Un lenguaje interpretado o de script, se entiende como aquel que es ejecutado por medio de un intérprete, por lo tanto, cuando se habla de python se habla de un lenguaje interpretado, dinámico, fuertemente tipado, multiplataforma, y orientado a objetos. Todo esto en palabras resumidas de su creador, Guido Van Rossum en los años 90 y a su vez inspirado en el grupo "Monty Python" que es un lenguaje similar a Perl, pero con una sintaxis muy simple y que favorece a un código legible (Duque, 2011, p. 7)

2.2.7. Anaconda.

Es un sistema y entorno de gestión de paquetes de código abierto, sistema de administración que se ejecuta en Windows, Mac OS y Linux. Conda instala, ejecuta y actualizar rápidamente paquetes y sus dependencias. Conda crea, guarda, carga y cambia fácilmente entre entornos en el equipo local. fue creado para programas Python, pero puede empaquetar y distribuir software para cualquier idioma.

Como administrador de paquetes, Conda ayuda a encontrar e instalar paquetes. sí se necesita un paquete que requiere una versión diferente de Python, no es necesario cambiar a un entorno

diferente, porque Conda también administra diferentes medios con tan solo unos pocos comandos, puede configurar un entorno totalmente separado para ejecutar esa versión diferente de Python, mientras continúa ejecutándose su versión habitual de Python en su entorno normal (Anaconda, 2017).

2.2.8. Yolo (You Only Look Once).

Enmarca un concepto puntual puesto que se trata de una arquitectura para la detección de objetos muy precisa y evidentemente rápida como describe su principal creador Joseph Redmond (2016, p. 2-8), por ser una arquitectura muy rápida, se vuelve la más idónea para la detección sobre vídeo en tiempo real. Está constituido principalmente por una red neuronal convolucional para predecir simultáneamente varios cuadros delimitadores y las probabilidades de las clases de objetos que delimitan esos cuadros.

Como se puso en manifiesto, su funcionamiento se basa en una red neuronal convolucional, que usa características de la imagen de entrada para trazar los cuadros delimitadores, con esto se logra que la red neuronal reaccione a cada objeto que se muestra en la imagen administrada por el usuario. La imagen captada por la cámara es dividida en una malla de $S \; x \; S$ de tal forma que, si un objeto ingresa en la celda de la malla, esa celda es responsable de detectar dicho objeto. (Redmon, 2016, p. 2-8)

Para un mejor entendimiento del proceso que realiza Yolo, imagine que Yolo detecta jugadores y balones de futbol a partir de una imagen determinada. A continuación se desglosa como obtener la ilustración 3-2 (B) a partir de la ilustración 3-2 (A). (DataCamp, 2022)

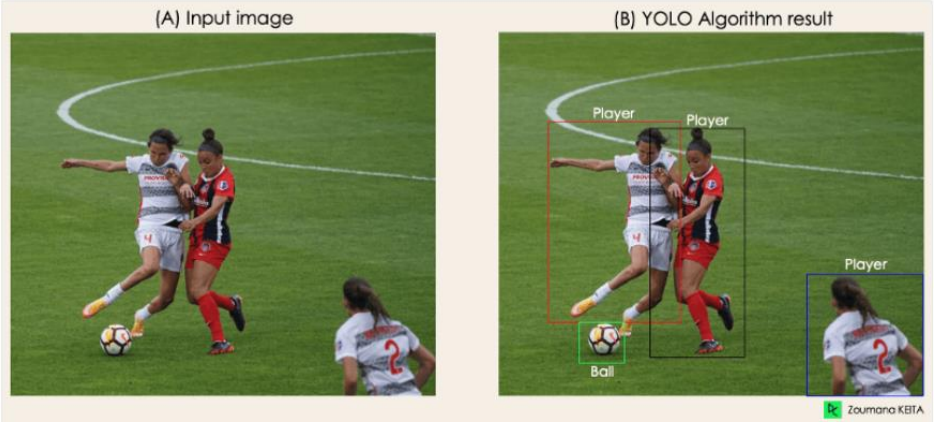


Ilustración 3-2: (A) Imagen de entrada, (B) Imagen de salida procesada por el algoritmo de Yolo

Fuente: (DataCamp, 2022)

Hay cuatro enfoques principales en el funcionamiento del algoritmo Yolo:

- ❖ Bloques Residuales.
- ❖ Regresión del cuadro delimitador.
- ❖ Intersección sobre sindicatos o pagaré para abreviar.
- ❖ Supresión no máxima.

2.2.8.1. Bloques residuales.

En primer lugar, el primer proceso divide la ilustración 4-2 (A) en celdas de mallas $S\ x\ S$ de igual forma y tamaño, para ejemplificar en nuestro caso S es 4 como se muestra en la ilustración 4-2 (C), y aquí el enfoque principal, porque cada celda de la malla es responsable de localizar y predecir el objeto que contiene en conjunto con su grado de confianza. (DataCamp, 2022)

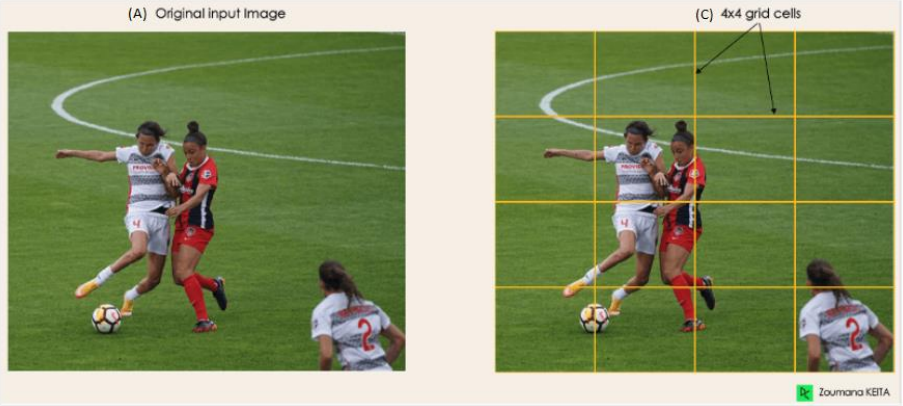


Ilustración 4-2: (A) Imagen de entrada, (C) Bloques residuales en malla de $4\ x\ 4$

Fuente: (DataCamp, 2022)

2.2.8.2. Regresión del cuadro delimitador.

Como segunda parte del proceso, inmediatamente se determinan rectángulos funcionando como cuadros delimitadores para resaltar los objetos de la imagen, para ello se delimitarán tantos rectángulos como objetos existan en la imagen proporcionada. Entonces, Yolo es el encargado de determinar cada uno de los atributos de los cuadros delimitadores, haciendo uso de un único módulo de regresión en el formato $Y = [pc, bx, by, bh, bw, c1, c2]$, con Ycomo la representación final de cada cuadro delimitador. Siendo esto muy importante en especial en la fase de entrenamiento.

Para entender de mejor manera, aquí una explicación más profunda. El parámetro pc se encarga de puntuar en base a probabilidades el valor de confianza del objeto que dice ser contenido en la celda. La ilustración 5-2 (D) tiene recuadros de dos colores diferentes, en las celdas amarillas no se encuentra ningún objeto por lo que el valor de confianza es 0, mientras que en las celdas de color rojo existen objetos a detectar que son de interés, por tano, cada recuadro contiene un segmento del objeto en común con cierto grado de confianza. Esto por su puesto se ve de mejor manera en la ilustración 5-2 (E) (DataCamp, 2022).

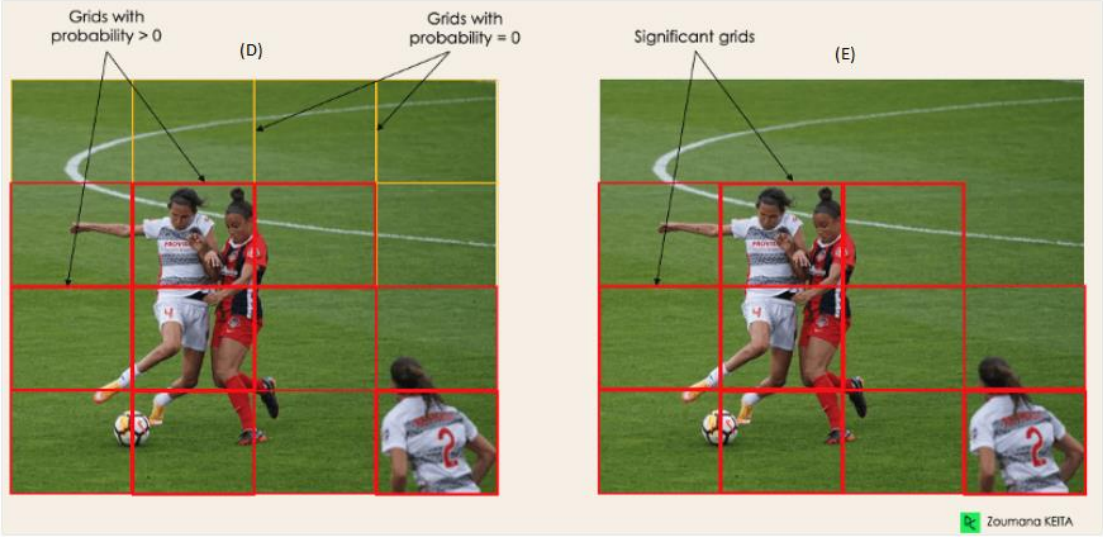


Ilustración 5-2: (D) Regresión con cuadros delimitadores, (E) Eliminación de los cuadros delimitadores con confianza 0

Fuente: (DataCamp, 2022)

Siguiendo con la explicación, bx y by son coordenadas en los puntos (x, y) en el centro del cuadro delimitador con respecto a la celda de la malla envolvente. Por su parte bh y bw se encargar de definir la altura y el ancho del cuadro delimitador respectivamente, restando $c1$ y $c2$ que permiten entender las clases latentes, en esta caso jugador y pelota, en este caso se puede tener tantas clases se quiera según el uso que se presente. Ver la ilustración 6-2.

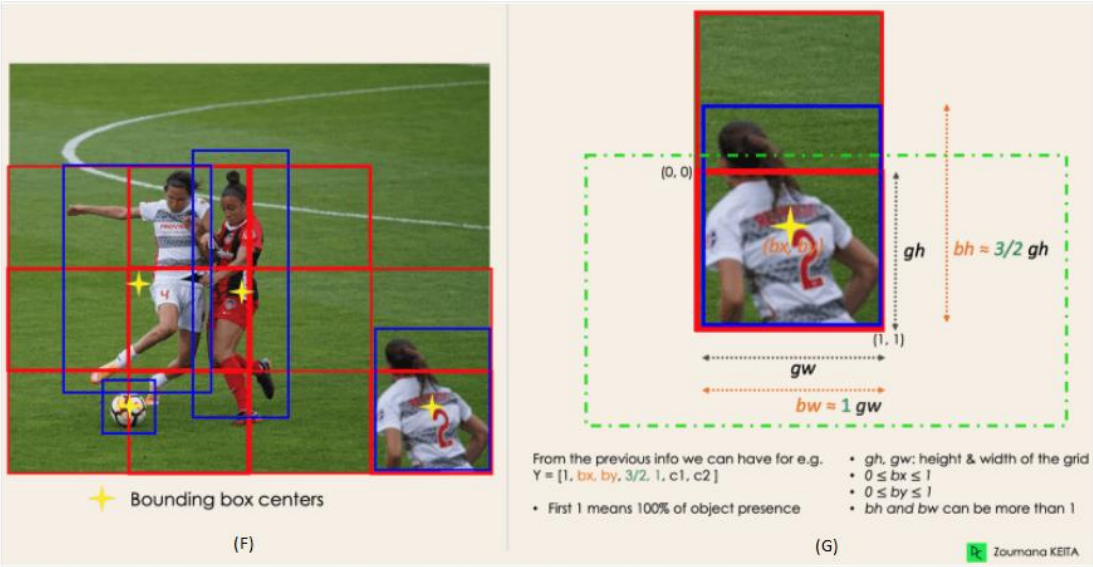


Ilustración 6-2: (F) Centro del recuadro delimitador, (G) Parámetros detallados en un objeto en específico.

Fuente: (DataCamp, 2022)

2.2.8.3. Intersección sobre sindicatos o pagarés.

Mediante el proceso es habitual encontrar más de un objeto por imagen y por ende múltiples celdas de la malla para poder predecir, aquí es importante recalcar que no todas las celdas contienen objetos relevantes, por consecuencia, se deben establecer pagarés que no es más que un valor comprendido entre 0 y 1 para descartar cada celda de la malla y mantener únicamente las celdas con contenido relevante. Comprendido de la siguiente manera:

- El usuario es quien se encarga de definir el umbral de pagaré, un valor aleatorio mayor a 0 y menor que 1, por ejemplo 0.5.
- Yolo está en la capacidad de poder calcular el IOU (Intersection Over Union) que de forma simple se entiende como la relación entre las regiones que corresponden a los verdaderos positivos y falsos positivos. Dicho de otra manera, es el área de intersección dividida por el área de unión.
- Con lo anterior establecido, el paso final es solo quitar las celdas cuyo pagaré sea menor al establecido y dejar únicamente los que sean mayores o igual al pagaré.

La ilustración 7-2 comprende el proceso de selección de celda, como se aprecia el objeto originalmente tiene 2 candidatos para la celda, luego solo se selecciona la celda 2.

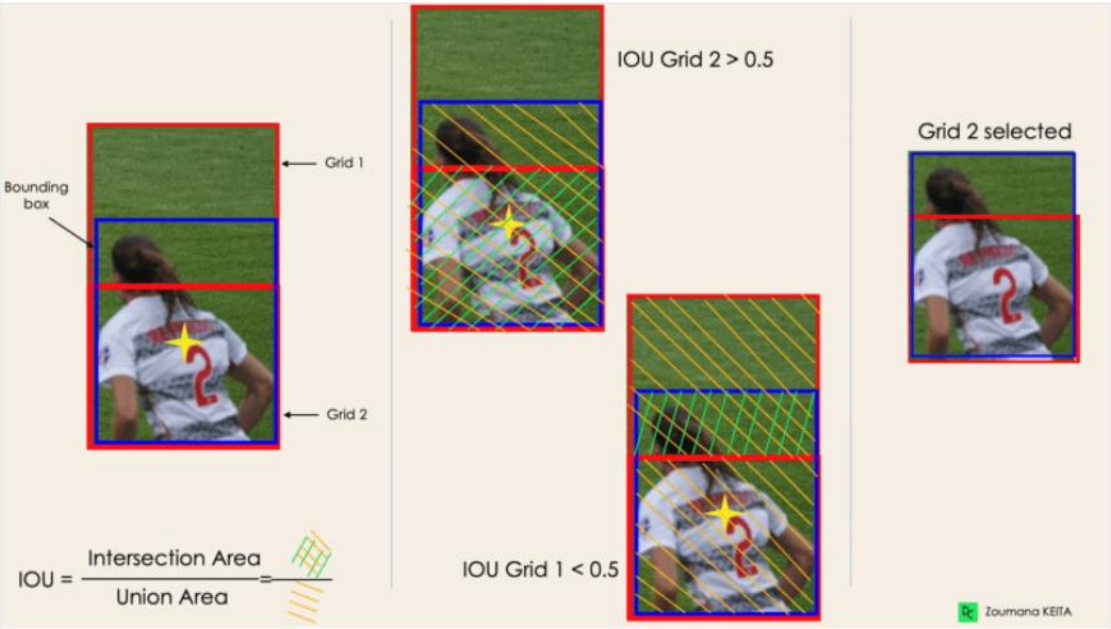


Ilustración 7-2: Selección del recuadro en base al pagaré establecido.

Fuente: (DataCamp, 2022)

2.2.8.4. Supresión no máxima.

No siempre se obtiene la mejor precisión estableciendo solo un nivel de umbral para el pagaré, para tener mayor eficiencia se recomienda mantener solo las cajas con mayor puntuación de probabilidad de detección dejando fuera las que pueden incluir ruido. (DataCamp, 2022).

2.2.8.5. Arquitectura Yolo.

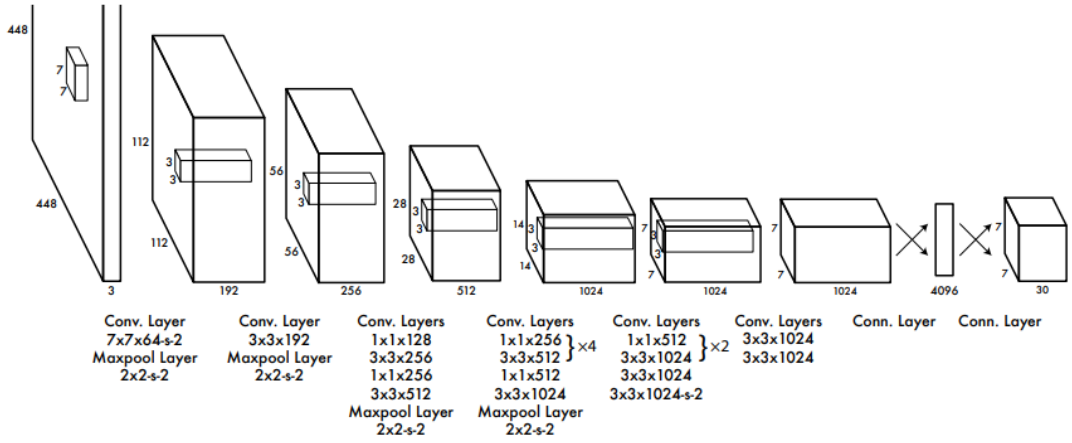


Ilustración 8-2: Arquitectura Yolo.

Fuente: (Redmon, 2016, p. 3)

Se detalla entonces la ilustración 8-2. El primer paso es cambiar el tamaño de la imagen que en un principio se consideraba de $448\ x\ 448$ pixeles antes de pasarla por la red convolucional. El segundo paso es aplicar una convolución $1\ x\ 1$ reduciendo el número de canales y al instante una convolución de $3\ x\ 3$ para tener una salida cuboidal. Por consiguiente, un tercer paso con una función de activación bajo el capó ReLU, exceptuando la capa final que usa una función lineal. Finalmente se emplean algunas técnicas adicionales, entre ellas la normalización de lotes y el abandono respectivamente, evitando que el modelo se sobreajuste con una regularización del modelo (Redmon, 2016, p. 2-8).

2.2.8.6. Versiones.

Año 2015, fecha en la que parte y se fundan las bases de lo que hoy es un algoritmo robusto, accesible, pasando por diferentes versiones hasta la actualidad, tal como se muestra en la ilustración 9-2.

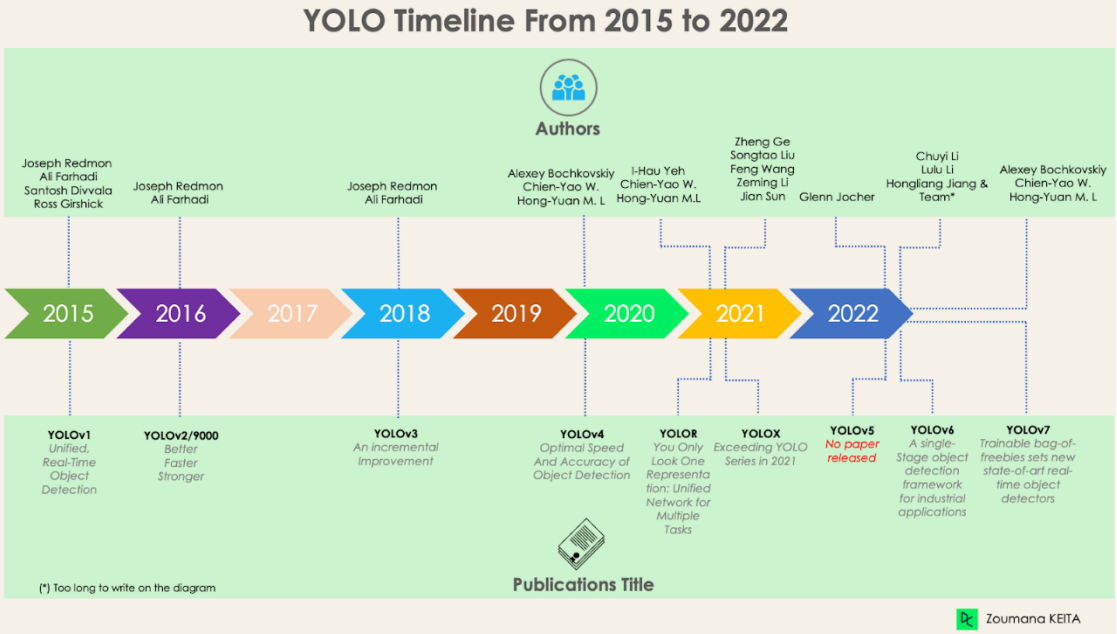


Ilustración 9-2: Línea de tiempo de Yolo.

Fuente: (DataCamp, 2022)

2.2.8.6.1. Yolo v1.

Por ser considerada la primera versión de Yolo, muchos consideraban este modelo como una manera de realizar un cambio a la manera de detectar objetos debido a su capacidad, rapidez y eficiencia.

Las dificultades no se hicieron esperar puesto que eran notorias, si se le daba un conjunto de imágenes que contiene objetos pequeños sus falencias eran obvias, debido a que cada celda de las mallas tenía como objetivo detectar su propio objeto de manera individual (Redmon, 2016, p. 2-8).

2.2.8.6.2. Yolo v2.

Conocido también como Yolo 9000, mejor, más rápido, más fuerte. Tenía la capacidad de reconocer más de 9000 categorías de objetos, aquí se evalúa la Precisión media de la métrica (mAP) para evaluar el modelo de aprendizaje obteniendo los 78.6 de precisión media de métrica sin dejar de funcionar relativamente rápido y ejecutándose en tiempo real (Redmon & Farhadi, 2016, p. 2-4).

YOLOv2 utiliza directamente una entrada 448×448 pixeles de mayor resolución en lugar de 224×224 pixeles , lo que hace que el modelo ajuste su filtro para que funcione mejor en imágenes de mayor resolución. Este enfoque aumentó la precisión en un 4% mAP, después de haber sido

entrenado durante 10 épocas. Simplifica el problema reemplazando las capas totalmente conectadas con cajas de anclaje. Este enfoque disminuyó ligeramente la precisión, pero mejoró la recuperación del modelo en un 7%, lo que da más margen de mejora (Redmon & Farhadi, 2016, p. 2-4).

2.2.8.6.3. Yolo v3.

Yolo sufre más cambios implementados por su creado (Redmon & Farhadi, 2018, p. 2-5). Los cambios incluyen principalmente una nueva arquitectura de red: Darknet-53. Esta es una red neuronal 106, con redes de muestreo ascendente y bloques residuales. Es mucho más grande, más rápido y preciso en comparación con Darknet-19, que es la columna vertebral de YOLOv2. Esta nueva arquitectura ha sido beneficiosa en muchos niveles:

- YOLOv3 utiliza un modelo de regresión logística para predecir la puntuación de objetividad para cada cuadro delimitador.
- En lugar de utilizar softmax como se realiza en YOLOv2, se han introducido clasificadores logísticos independientes para predecir con precisión la clase de los cuadros delimitadores. Esto es incluso útil cuando se enfrentan a dominios más complejos con etiquetas superpuestas (por ejemplo, Persona → Jugador de fútbol). El uso de un softmax limitaría cada caja a tener una sola clase, lo que no siempre es cierto.
- YOLOv3 realiza tres predicciones a diferentes escalas para cada ubicación dentro de la imagen de entrada para ayudar con el muestreo ascendente de las capas anteriores. Esta estrategia permite obtener información semántica detallada y más significativa para una imagen de salida de mejor calidad.

2.2.8.6.4. Yolo v4.

Como en cada versión la idea es tener mayor precisión, mayor velocidad y una detección optima. La ilustración 10-2 muestra como YOLOv4 supera a YOLOv3 en fotogramas por segundo (FPS) y velocidad en un 10% y 12% respectivamente (Bochkovskiy, Wang & Hong-Yuan, 2020, p. 3-10)

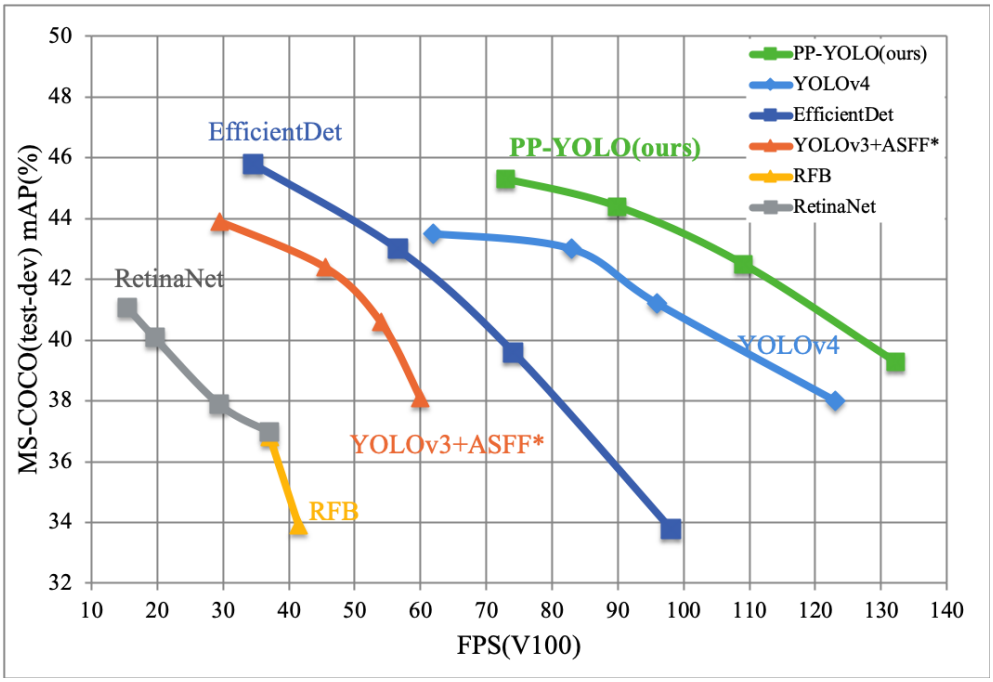


Ilustración 10-2: Comparación de Yolo v4 con Yolo v3 y otras tecnologías para la detección de objetos de última generación.

Fuente: (Bochkovskiy, Wang & Hong-Yuan, 2020, p. 1)

Esta arquitectura, en comparación con YOLOv3, agrega la siguiente información para una mejor detección de objetos:

- El bloque de agrupación piramidal espacial (SPP) aumenta significativamente el campo receptivo, segrega las características de contexto más relevantes y no afecta la velocidad de la red.
- El bloque de agrupación piramidal espacial (SPP) aumenta significativamente el campo receptivo, segrega las características de contexto más relevantes y no afecta la velocidad de la red.
- El bloque de agrupación piramidal espacial (SPP) aumenta significativamente el campo receptivo, segrega las características de contexto más relevantes y no afecta la velocidad de la red.
- Realice una selección óptima de hiperparámetros utilizando algoritmos genéticos.

2.2.8.6.5. Yolo v5.

Aquí es muy importante enfatizar que es la primera versión de Yolo que no cuenta con un trabajo de investigación publicado y además la primera también en implementarse en Pytorch en lugar de Darknet.

Una de las principales mejoras en la arquitectura YOLOv5 es la integración de la capa Focus, representada por una sola capa, que se crea reemplazando las tres primeras capas de YOLOv3. Esta integración redujo el número de capas y el número de parámetros y también aumentó la velocidad hacia adelante y hacia atrás sin ningún impacto importante en el mAP. (DataCamp, 2022)

La ilustración 11-2 compara el tiempo de entrenamiento de Yolo v5 contra Yolo v4

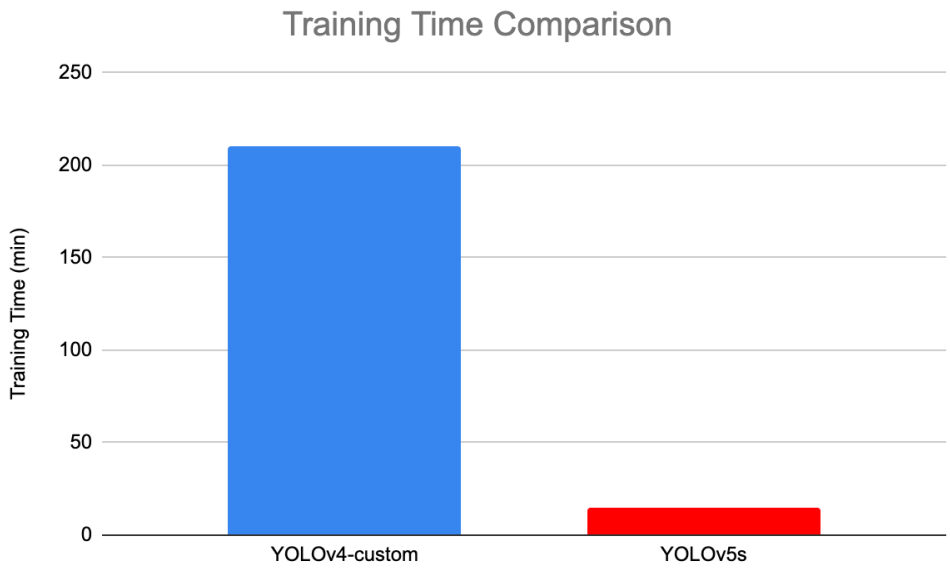


Ilustración 11-2: Tiempo de entrenamiento menor dado por Yolo v5 en comparación con Yolo v4.

Fuente: (DataCamp, 2022)

2.2.8.6.6. Yolo v6.

En términos de implementación Yolo v6 admite la implementación utilizando diferentes métodos como GPU, TensorRT, ARM Y ONNX. Es un marco de detección de objetos de una sola etapa que se centra únicamente en aplicaciones industriales con un diseño eficiente de hardware y un mejor rendimiento que sus versiones anteriores.

Yolo v6 ha sido una adición reciente a la larga serie de algoritmos Yolo, el proyecto se encuentra en constante desarrollo a medida que más personas contribuyan y aporten nuevas características y estabilidad al modelo (Revca, 2022).

2.2.9. Darknet.

Es una red neuronal convolucional y una red troncal para la detección de objetos que utiliza DarkNet-53. Emplea una estrategia CSPNet (Red parcial de etapa cruzada) para particionar el mapa de entidades de la capa base en dos partes y, a continuación, las combina a través de una jerarquía entre etapas (Bochkovskiy, Wang & Hong-Yuan, 2020, p. 3-10).

2.2.10. Coco Datasets.

COCO (Common Objects in Context) dadas sus siglas en inglés, comprende a un conjunto de datos de datos de referencia para la detección de objetos administrando funcionalidades como:

- Segmentación de objetos.
- Reconocimiento del contexto.
- Segmentación de identidades mediante superpixel.
- Más de 300000 imágenes de las cuales 220000 están categorizadas.
- 1.5 millones de instancias de objetos.
- 80 categorías de objetos.
- 91 clasificaciones de entidades.
- 5 posibles títulos por imagen.
- 250000 personas con punto de interés.

En forma de resumen, se necesita un conjunto de datos entrenados previamente para disminuir la latencia al momento de detectar objetos, este componente de datos será administrado por COCO y sus funcionalidades aplicativas propuestas por (Yi Ling, Maire & Belongie, 2015, p. 2-8).

2.2.11. Redes Neuronales.

En perspectiva se comprende a una red neuronal como un conjunto de neuronas artificial conectadas entre y enviándose señales continuamente efectuando una comunicación plena, para cumplir con un propósito en común. Según el grado de complejidad se pueden armar diferentes topologías y según el método de aprendizaje se pueden crear bucles dentro de la red neuronal, que permitan el almacenamiento de información a lo largo del tiempo (Calvo, 2017).

De entre sus variabilidades tenemos la red neuronal monocapa que corresponde a una red neuronal más simple compuesta por una capa de neuronas qué proyectan las entradas a una capa

de neuronas de salida dónde se realizan los diferentes cálculos, para entender este proceso analice la ilustración 12-2.

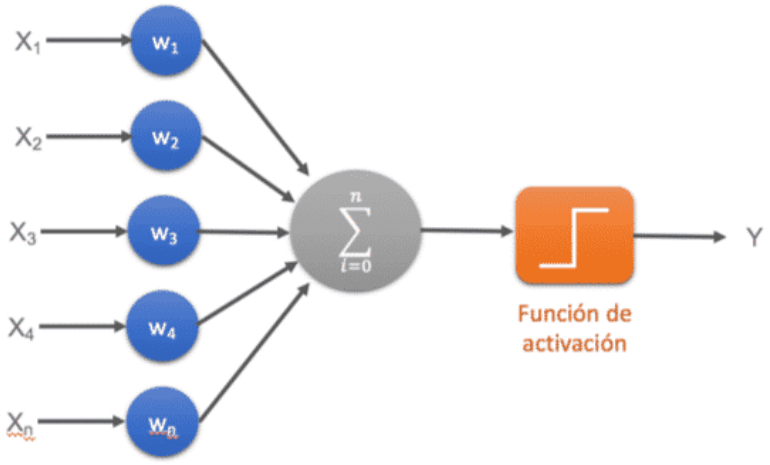


Ilustración 12-2: Red neuronal monocapa.

Fuente: (Calvo, 2017)

Otra configuración de las redes neuronales se presenta en la ilustración 13-2 y se basa en la conexión multicapa que dispone de un conjunto de capas intermedias o capas ocultas entre la capa de entrada y la capa de salida, dependiendo además del número de conexiones esta red puede estar parcial o totalmente conectada.

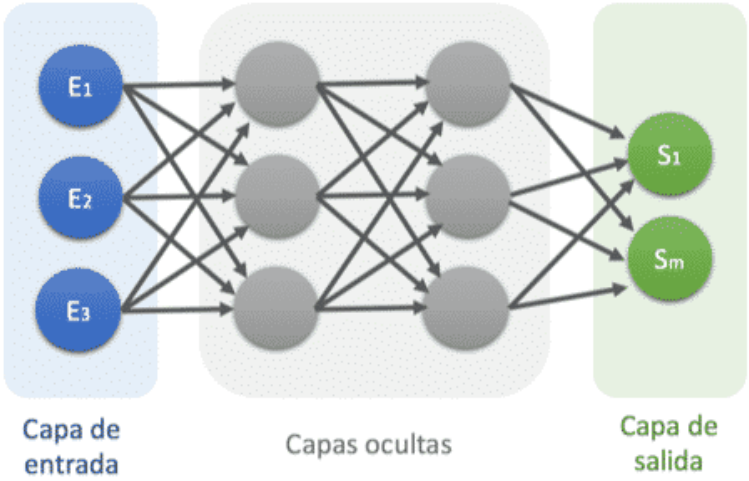


Ilustración 13-2: Red neuronal multicapa.

Fuente: (Calvo, 2017)

2.2.12. Redes Convolucionales.

Existe una diferencia entre las redes neuronales convolucionales y las redes neuronales multicapa y principalmente proviene del hecho de que cada neurona solo se conecta o se especializa a un subgrupo de ellas, en lugar de a todas las capas de la siguiente capa. Con esto se logra reducir el número de neuronas requeridas y la complejidad computacional requerida para ejecutarlas de la forma que se pude identificar en la ilustración 14-2.

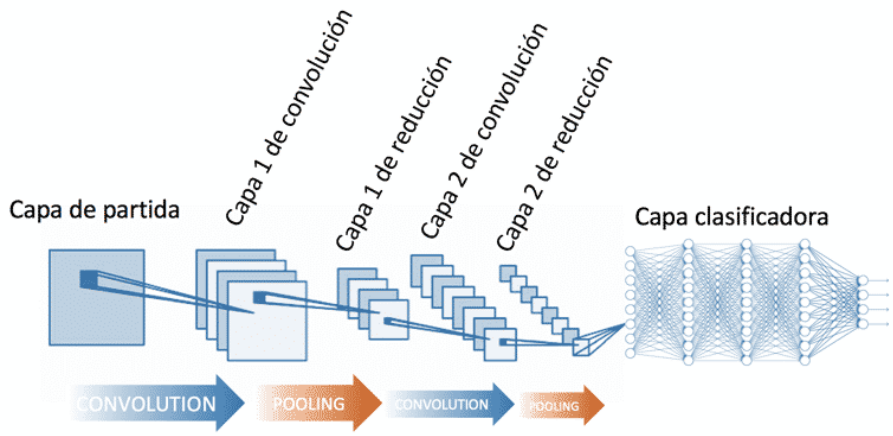


Ilustración 14-2: Red neuronal convolucional.

Fuente: (Calvo, 2017)

2.2.13. Open CV.

OpenCV (Open Source Computer Vision Library) es una biblioteca de software de visión artificial y aprendizaje automático de código abierto. OpenCV fue construido para proporcionar una infraestructura común para aplicaciones de visión artificial y para acelerar el uso de la percepción de la máquina en los productos comerciales. Al ser un producto con licencia Apache 2, OpenCV facilita a las empresas la utilización y modificación del código.

La biblioteca tiene más de 2500 algoritmos optimizados, que incluyen un conjunto completo de algoritmos de visión por computadora y aprendizaje automático clásicos y de última generación. Estos algoritmos se pueden utilizar para detectar y reconocer caras, identificar objetos, clasificar acciones humanas en videos, rastrear movimientos de cámara, rastrear objetos en movimiento, extraer modelos 3D de objetos, producir nubes de puntos 3D de cámaras estéreo, unir imágenes para producir una imagen de alta resolución de una escena completa, encontrar imágenes similares de una base de datos de imágenes, eliminar ojos rojos de imágenes tomadas con flash, Seguir los movimientos oculares, reconocer paisajes y establecer marcadores para superponerlos

con realidad aumentada, etc. OpenCV tiene más de 47 mil personas de comunidad de usuarios y un número estimado de descargas que supera los 18 millones. La biblioteca se utiliza ampliamente en empresas, grupos de investigación y por organismos gubernamentales (OpenCV, 2023).

2.2.14. Pytorch.

Pytorch con su biblioteca de código abierto basada en torch permite usarlo frente a aplicaciones como visión por computadora y procesamiento de lenguaje natural, es por esto por lo que se puede implementar un detector de objetos basados en You Only Look Once (YOLO), que utiliza características aprendidas por una red neuronal convolucional profunda para detectar un objeto (VMWare, 2022).

Pytorch proporciona características de alto nivel, como computación por tensores parecidos a los de Numpy, con una aceleración fuerte a través de unidades de procesamiento gráficos (GPU) y redes neuronales profundas construidas en un sistema de diferenciación automática de base de datos. Utiliza un tensor (torch.tensor) para operar y almacenar arreglos de número rectangulares, homogéneos y multidimensionales con la ventaja de que se pueden operar sobre una GPU de NVIDIA (Analytics, 2023).

PyTorch utiliza un paradigma imperativo / ansioso. Es decir, cada línea de código necesaria para construir un gráfico define un componente de ese gráfico. Podemos realizar cálculos de forma independiente en estos componentes, incluso antes de que su gráfico se construya por completo. Esto se denomina metodología de "definir por ejecución"(Analytics, 2023).

Para su aprendizaje profundo, Pytorch cuenta con 5 elementos importantes que se tienen que conocer para comenzar a usarlo.

2.2.14.1. Tensores Pytorch.

Los tensores no son más que matrices multidimensionales. Los tensores en PyTorch son similares a los arreglos de numpy, con la adición de que los tensores también se pueden usar en una GPU. PyTorch admite varios tipos de tensores y su forma de importarlos es como se ve en la ilustración 15-2.

```
# import pytorch
import torch

# define a tensor
torch.FloatTensor([2])
```

```
 2
[torch.FloatTensor of size 1]
```

Ilustración 15-2: Matriz unidimensional simple.

Fuente: (Analytics, 2023)

2.2.14.2. Operaciones Matemáticas.

Al igual que con numpy, es crucial que una biblioteca de computación científica tenga implementaciones eficientes de funciones matemáticas. PyTorch ofrece una interfaz similar, con más de 200 operaciones matemáticas que se pueden utilizar. (Analytics, 2023). A continuación, se muestra un ejemplo en la ilustración 16-2 de una operación de adición simple en PyTorch:

```
a = torch.FloatTensor([2])
b = torch.FloatTensor([3])

a + b
```

```
 5
[torch.FloatTensor of size 1]
```

Ilustración 16-2: Operaciones matemáticas simples.

Fuente: (Analytics, 2023)

2.2.14.3. Módulo Autograd.

PyTorch utiliza una técnica llamada diferenciación automática. Por ejemplo, se tiene un grabador que registra qué operaciones se han realizado, y luego lo reproduce hacia atrás para calcular los degradados. Esta técnica es especialmente poderosa cuando se construyen redes neuronales, ya que ahorramos tiempo en una época, al calcular la diferenciación de los parámetros en el propio paso hacia adelante (PyTorch, 2023). Vea la ilustración 17-2.

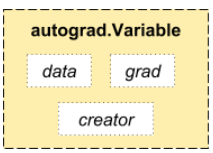


Ilustración 17-2: Autograd Variable.

Fuente: (Pytorch, 2023)

2.2.14.4. Módulo Optim.

Torch.optim presentado en la ilustración 18-2, es un módulo que implementa varios algoritmos de optimización utilizados para construir redes neuronales. La mayoría de los métodos de uso común ya son compatibles, por lo que se tiene que construir desde cero (Analytics, 2023).

```
optimizer = torch.optim.Adam(model.parameters(), lr=learning_rate)
```

Ilustración 18-2: Optimizador por módulo torch.optim.

Fuente: (Analytics, 2023)

2.2.14.5. Módulo NN.

PyTorch autograd facilita la definición de gráficos computacionales y la toma de gradientes, pero el autograd sin procesar puede ser un nivel demasiado bajo para definir redes neuronales complejas. Aquí es donde el módulo NN puede ayudar.

El paquete NN define un conjunto de módulos, que se puede considerar como una capa de red neuronal que produce salida de entrada y puede tener algunos pesos entrenables de tal forma que son llamados siempre como se manifiesta en la ilustración 19-2 (Analytics, 2023).

```
import torch

# define model
model = torch.nn.Sequential(
 torch.nn.Linear(input_num_units, hidden_num_units),
 torch.nn.ReLU(),
 torch.nn.Linear(hidden_num_units, output_num_units),
)
loss_fn = torch.nn.CrossEntropyLoss()
```

Ilustración 19-2: Salida convolucional NN.

Fuente: (Analytics, 2023)

2.2.15. Pandas.

Pandas es un paquete Python de código abierto que se usa para tareas de ciencia de datos, análisis de datos y aprendizaje automático. Está construido sobre otro paquete llamado Numpy, que proporciona soporte para matrices multidimensionales. Pandas funciona bien con muchos otros módulos de ciencia de datos dentro del ecosistema de Python y, por lo general, se incluye en todas las distribuciones de Python, desde las que vienen con su sistema operativo hasta las distribuciones de proveedores comerciales como ActivePython de ActiveState (Pandas Org, 2023).

Línea de tiempo

- 2008: Comienza el desarrollo de pandas.
- 2009: Pandas se convierte en código abierto.
- 2012: Se publica la primera edición de Python for Data Analysis.
- 2015: pandas se convierte en un proyecto patrocinado por NumFOCUS.
- 2018: Primer sprint de desarrollador principal en persona.

2.2.16. Seaborn.

Seaborn es una biblioteca para hacer gráficos estadísticos en Python. Se basa en matplotlib y se integra estrechamente con las estructuras de datos de pandas. Seaborn le ayuda a explorar y comprender sus datos. Sus funciones de trazado presentadas en la ilustración 20-2 operan en marcos de datos y matrices que contienen conjuntos de datos completos e internamente realizan el mapeo semántico y la agregación estadística necesarios para producir gráficos informativos (Waskom, 2012).

```
# Import seaborn
import seaborn as sns

# Apply the default theme
sns.set_theme()

# Load an example dataset
tips = sns.load_dataset("tips")

# Create a visualization
sns.relplot(
    data=tips,
    x="total_bill", y="tip", col="time",
    hue="smoker", style="smoker", size="size",
)
```

Ilustración 20-2: Exportación de Seaborn

Fuente: (Waskom, 2012)

2.2.17. Matplotlib.

Matplotlib es una biblioteca completa para crear archivos estáticos, animados, y visualizaciones interactivas en Python como los que se plasman en la ilustración 21-2. Matplotlib hace las cosas fáciles y difíciles posibles (Matplotlib, 2022).

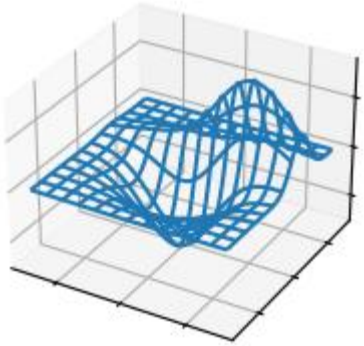


Ilustración 21-2: Gráfica 3D obtenida de Matplotlib.

Fuente: (Matplotlib, 2022)

2.2.18. Numpy.

NumPy es un proyecto de código abierto que tiene como objetivo habilitar la computación numérica con Python. Fue creado en 2005, basándose en el trabajo inicial de las bibliotecas Numeric y Numarray. NumPy siempre será 100% software de código abierto, libre para todos y publicado bajo los términos liberales de la licencia BSD modificada (Numpy, 2005).

NumPy es el paquete fundamental para la computación científica en Python. Es una biblioteca de Python que proporciona un objeto de matriz multidimensional, varios objetos derivados (como matrices y matrices enmascaradas) y un Surtido de rutinas para operaciones rápidas en arreglos de discos, incluyendo matemático, lógico, manipulación de formas, ordenación, selección, E/S, Transformadas discretas de Fourier, álgebra lineal básica, estadística básica operaciones, simulación aleatoria y mucho más (Numpy, 2018).

2.2.19. Pillow.

Es una biblioteca que suministra una amplia compatibilidad con formatos de archivos para una interpretación interna más eficiente gracias a que agrega capacidades de procesamiento de imágenes bastante potentes. Principalmente está diseñada para un acceso rápido a los datos almacenados en unos pocos formatos básicos de pixeles, por esto qué, generalmente está diseñada para el tratamiento de imágenes (Pillow ORG, 2023).

2.2.20. PyYaml.

Tiene la funcionalidad de serializar datos que hacen referencia a formatos python pero que de cierto modo es entendible para los humanos y capaz de interactuar con lenguajes de scripting, funcionando como analizador y emisor YAML de python.

Este analizador es capaz de soportar completamente Unicode, API de extensión y mensajes de erros sensibles. Permite etiquetas YAML estándar para proporcionar etiquetas específicas de python que permite referenciar un objeto python arbitrario (PyYaml, 2021).

2.2.21. Requests.

Esta biblioteca de solicitudes actúa como un estándar de facto para realizar solicitudes HTTP en Python. Eliminando las complejidades de realizar solicitudes detrás de una API simple, que permite concentrarse e interactuar con los servicios y consumir datos en su aplicación (Requests, 2021).

2.2.22. SciPy.

SciPy es una colección de algoritmos matemáticos y con funciones basadas en la extensión NumPy de Python. Añade potencia significativa para la sesión interactiva de Python al proporcionar el usuario con comandos y clases de alto nivel para manipular y Visualización de datos. Con SciPy, una sesión interactiva de Python se convierte en un entorno de procesamiento de datos y creación de prototipos de sistemas que rivaliza con sistemas, como MATLAB, IDL, Octave, R-Lab y SciLab.

El beneficio adicional de SciPy en Python, es tener un potente lenguaje de programación disponible para su uso en el desarrollo Programas sofisticados y aplicaciones especializadas (SciPy, 2023).

2.2.23. Tqdm.

Tqdm es una biblioteca en Python que se utiliza para crear medidores de progreso o barras de progreso como se pone en manifiesto en la ilustración 22-2. tqdm obtuvo su nombre del nombre árabe ***taqaddum*** que significa "progreso". La implementación de tqdm se puede hacer sin esfuerzo en los bucles, funciones o incluso Pandas. Las barras de progreso son bastante útiles en Python porque:

- Se puede ver si el Kernal sigue funcionando.
- Las barras de progreso son visualmente atractivas a la vista.
- Proporciona tiempo de ejecución del código y tiempo estimado para que el código se complete, lo que ayudaría mientras se trabaja en grandes conjuntos de datos. (TQDM ORG, 2023).

Ilustración 22-2: Barra de progreso TQDM

Fuente: (TQDM ORG, 2023)

2.2.24. iPython.

iPython proporciona un amplio kit de herramientas presentadas en la ilustración 23-2 para ayudarle a aprovechar al máximo el uso de Python interactivamente. (iPython, 2023).

Sus principales componentes son:

- Un potente shell interactivo de Python.
- Un kernel de Jupyter para trabajar con código Python en cuadernos de Jupyter y otras interfaces interactivas.

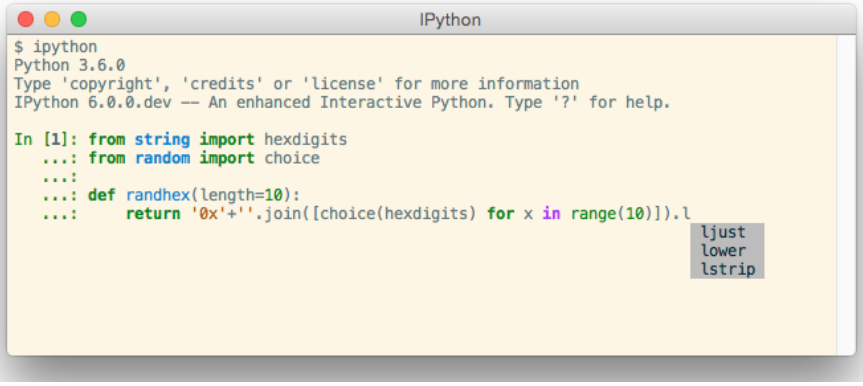


Ilustración 23-2: Configuración iPython

Fuente: (iPython, 2023)

2.2.25. Psutil.

PSUTIL (Process and System Utilities) es una biblioteca multiplataforma para recuperar información sobre procesos en ejecución y utilización del sistema (CPU, memoria, discos, red, sensores) en Python. Es útil principalmente para el monitoreo del sistema, el perfil y el proceso de limitación. recursos y gestión de procesos en ejecución. Implementa muchas funcionalidades ofrecidas por las herramientas clásicas de línea de comandos de UNIX como *ps*, *top*, *iotop*, *lsof*, *netstat*, *ifconfig*, *free* y otros (PyPi, 2023).

2.2.26. Telegram.

Telegram es mensajería basada en la nube con sincronización constante. Como resultado, se puede acceder a los mensajes desde diferentes dispositivos a la vez, incluyendo tablets y computadoras, y compartir un número ilimitado de fotos, videos y archivos (doc, zip, mp3, etc.) de hasta 2 GB cada uno. (Telegram org, 2023).

Telegram necesita menos de 100 MB en el dispositivo. Puede mantener todos los archivos multimedia en la nube sin necesidad de eliminar cosas, simplemente se borra la caché para liberar espacio. A parte de ser una aplicación muy liviana, su construcción en base de IA permite crear bots API, para que respondan de manera eficiente en tiempo real anclado a un código de ejecución en python.

2.2.27. Iriun.

Permite utilizar la cámara de cualquier teléfono conectado a la misma red como una cámara web inalámbrica en un PC, dando resultados como los indicados en la ilustración 24-2. (Iriun, 2020).

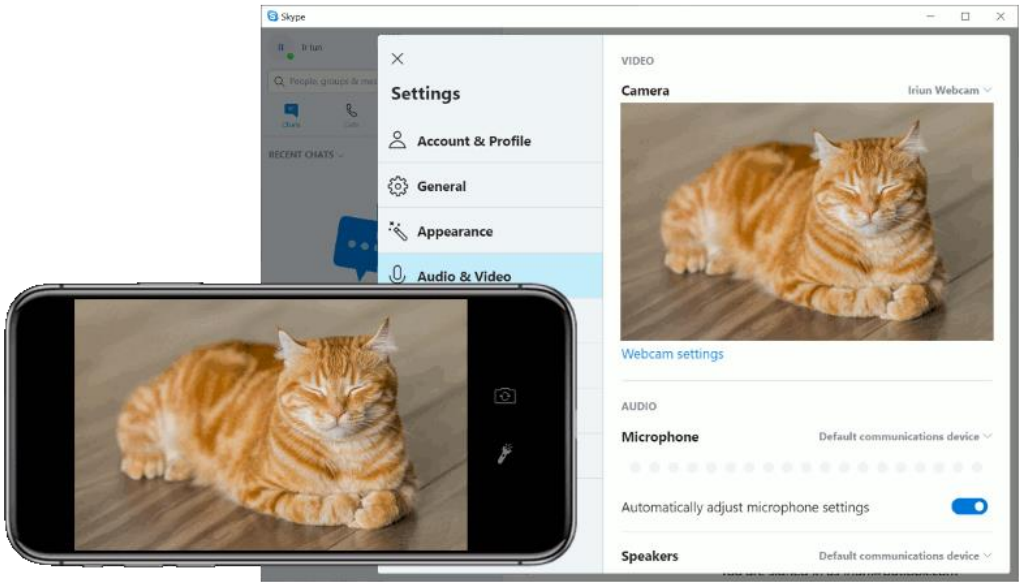


Ilustración 24-2: Funcionalidad de Iriun webcam.

Fuente: (Iriun, 2020)

2.2.28. MakeSense.

Es de código abierto y de uso gratuito bajo la licencia de GPLv3, admite formas de archivo de salida como Yolo, que en esencia admite múltiples tipos de etiquetas, rectángulos, líneas, polígonos y sobre todo no almacena las imágenes porque no se la envían a ninguna parte. Véase su interfaz en la ilustración 25-2. (MakeSense, 2023)

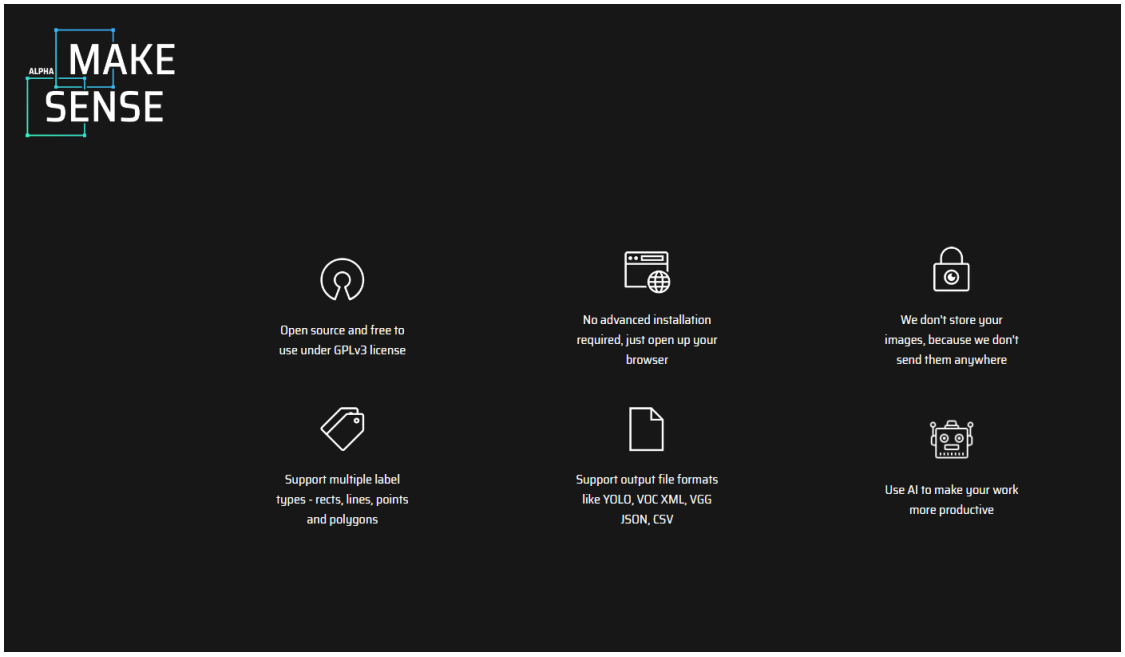


Ilustración 25-2: MakeSense etiquetas

Fuente: (MakeSense, 2023)

2.2.29. Modulo ESP3286

ESP32 es una serie de SoC (por sus siglas en inglés, System on Chip) (Ver la ilustración 26-2) y módulos de bajo costo y bajo consumo de energía creado por Espressif Systems. Esta nueva familia es la sucesora del famoso ESP3266 y su característica más notable es que además de Wi-Fi. En el mercado existen una infinidad de placas de desarrollo basadas en estos chips. Algunas especializadas en ciertas áreas como IoT, redes de sensores o aplicaciones de bajo consumo y otras de uso general. Lo cierto es que todas permiten implementar proyectos basados en ESP32 de forma muy simple, tal y como se lo hace con una placa Arduino cualquiera. (Last Minute Engineers, 2023)

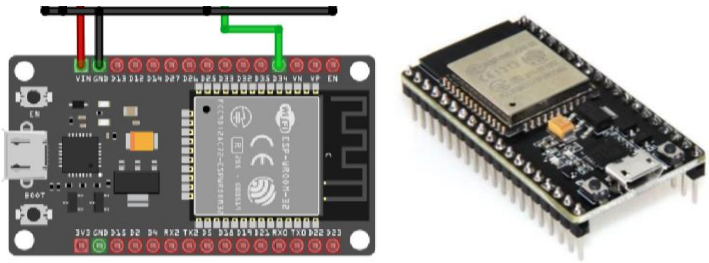


Ilustración 26-2: Módulo ESP 3286.

Fuente: (Last Minute Engineers, 2023)

2.2.30. Entorno IDE de Arduino

Un IDE es un entorno de programación como se ve en la ilustración 27-2 que ha sido empaquetado como un programa de aplicación; es decir, que consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (GUI). Además, en el caso de Arduino incorpora las herramientas para cargar el programa ya compilado en la memoria flash del hardware. Los programas de Arduino están compuestos por un solo fichero con extensión “ino”, aunque es posible organizarlo en varios ficheros. El fichero principal siempre debe estar en una carpeta con el mismo nombre que el fichero. (Last Minute Engineers, 2023)

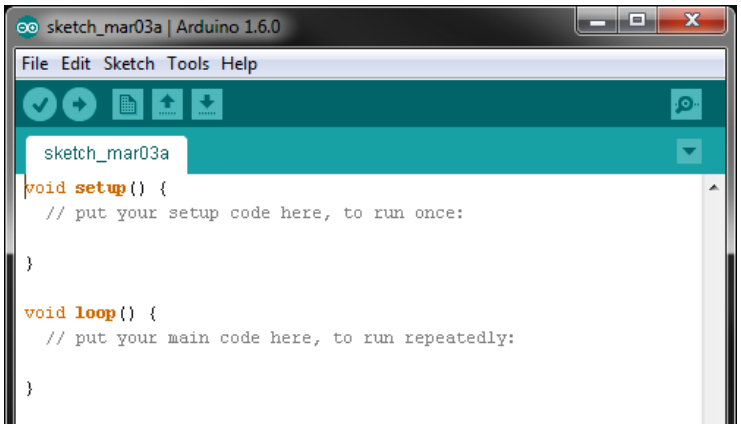


Ilustración 27-2: Entorno IDE

Fuente: (Last Minute Engineers, 2023)

2.2.31. Multisim

Como una herramienta de aprendizaje, Multisim™ conecta la teoría abstracta con señales concretas a través del diseño intuitivo, simulación interactiva y perfecta integración con hardware (National Instruments, 2023).

CAPITULO III

3. MARCO METODOLÓGICO.

En este capítulo se detallan todos los Procesos de forma ordenada y sistemática que son necesarios para poder implementar el prototipo, de manera que resulte fácil e intuitivo de seguir, para que tanto los usuarios como personas a fines puedan ejecutar el código y todas las herramientas de forma correcta.

3.1. Enfoque De Investigación.

El proyecto presentado basa la recolección de información y datos en una investigación cualitativa en la que se implementa un prototipo de emisión de alertas en zonas de peligro para el cuidado de niños de 1 a 5 años mediante la obtención de todos los conceptos que abarcan la visión artificial, para finalmente desplegar el prototipo para obtener los datos, procesarlos y analizar los resultados.

3.2. Nivel De La Investigación.

Aun dada la continua y amplia investigación sobre visión artificial, esta investigación se basa en una investigación exploratoria debido a que no existe artículos científicos que acompañen la funcionalidad de YOLO v5.

3.3. Diseño De La Investigación.

El presente trabajo está diseñado por una investigación cualitativa debido a que el diseño del prototipo usa la visión artificial como soporte principal. Y a su vez, se establece una conexión entre los datos recolectados para el entrenamiento de las redes neuronales y los datos observados por los elementos que componen la visión artificial.

3.3.1. Según la manipulación de la variable independiente.

El desarrollo del prototipo para la emisión de alertas utilizando visión por computadora se realiza de forma experimental, tanto el entrenamiento como la obtención del dataset necesario para la personalización de los pesos de cada red neuronal, para finalmente realizar la detección de los objetos en zonas de peligro.

3.3.2. Según las intervenciones en el trabajo de campo.

Es ideal aplicar un estudio transversal, planteando un tiempo de estudio prudente adecuado a las condiciones en el que interactúa el prototipo de detección de objetos por visión artificial y aprovecharlo para emitir las alertas, De este modo se identifica los resultados obtenidos en diferentes tiempos y escenarios.

3.4. Tipo De Estudio.

En vista que los datos son recolectados en base al problema que se desea solucionar y se hace contacto con el objeto motivo de estudio, usando técnicas de observación, encuestas, formularios y cuestionarios el trabajo de titulación expuesto hace uso de un tipo de estudio de campo, convirtiendo todos los procesos tanto teóricos y prácticos en algo de utilidad para la sociedad y su mejora en la calidad de vida, todos los conocimientos puesto en práctica son los mismos adquiridos durante todos los años de formación académica en la carrera de telecomunicaciones.

3.5. Población Y Planificación.

El prototipo de emisión de alertas basado en visión por computador tiene la finalidad de ser una herramienta de ayuda en el cuidado de niños de 1 a 5 años para reducir accidentes con objetos que en el entorno de crianza mimas resultan un verdadero peligro para los infantes. Adjunto al anexo F se puede evidenciar las respuestas obtenidas por diferentes tutores de crianza, padres y educadores parvularios de la encuesta realizada en base a la técnica Escala Likert con 5 respuestas posibles en un tiempo límite de 5 minutos.

3.6. Métodos, Técnicas E Instrumentos De Investigación.

El concepto de visión artificial comprende un campo amplio y con muchas temáticas que se pueden abarcar en diferentes tecnologías, el prototipo de emisión de alertas basado en visión por computador será validado y puesto a servicio de la sociedad en proceso de crianza, dado que su curiosidad los vuelve más propensos a sufrir lesiones debido a su inocencia. El prototipo en mención ha sido desarrollado en Python poniendo en escena todas las librerías necesarias para que el programa sea ejecutable en un computador con sistema operativo Windows, la aplicación

fue puesta a prueba con 4 niños en el rango de 1 a 5 años, 2 del sexo masculino y dos del sexo femenino.

3.6.1. Características de la población encuestada.

La población encuestada está compuesta por 3 tutores de crianza, un padre de familia primerizo, una niñera y una maestra de parvularia. Quienes estan encargados del cuidado de la integridad física y mental del infante.

3.6.2. Tipo de investigación.

Se usa una investigación cualitativa, pretendiendo entender cada uno de los fenómenos puesto en escena de forma natural sin que el investigador sea motivo de cambio de alguna variable que modifique o cambie el escenario. (Questionpro, 2023).

3.6.3. Métodos de la investigación.

Se detalla de forma respectiva los métodos usados para la investigación planteada, tanto el método analítico-sistemático como el método inductivo-deductivo.

3.7. Técnicas De La Investigación.

El trabajo actual emplea dos técnicas de investigación cualitativa a la par. En primera instancia se considera prudente una exhaustiva recolección bibliográfica con el fin de recolectar todo lo referente al tema tratado. Y, por otro lado, una investigación en base a la observación, entendiendo la información registrada en tecnologías ya existentes para analizarla de forma sistemática, para que finalmente desarrollado el prototipo, y ponerlo en prueba en base a una observación directa.

3.8. Selección De la Tecnología Para Ejecutar La Visión Artificial.

Como se ha expuesto con anterioridad, las tecnologías que se basan en visión por computador son muchas, aquí es donde se realiza el enfoque necesario en base a los requerimientos del sistema a realizar, para esto se debe tener en cuenta que el objetivo es poder reconocer objetos que mediante procesos de código serán identificados como peligro para la integridad física del infante. Es por esto por lo que la tabla 1-3a muestra las características fundamentales de dos

tecnologías empleadas tanto para la detección de objetos como para poder entrenar de manera profunda las redes convolucionales personalizando el prototipo solo a objetos de interés, con esto se logra reducir el costo computacional requerido para la visión artificial de manera que los resultados estén en un rango considerablemente bueno.

Tabla 1-3: Tabla descriptiva de las tecnologías para visión artificial

Yolo V5	PyTorch
Yolo compatible con python se compone de tres elementos principales: ❖ Columna vertebral: Una red neuronal convolucional que agrega formas y características de imagen en diferentes tipos de granularidad de imagen. YOLOv5 agrega la estructura Focus para realizar la operación de segmentación. Tomando la estructura de Yolov5 s como ejemplo, la imagen original de 640 × 640 × 3 se introduce en la estructura de enfoque, y la operación de corte se usa primero para formar un mapa de características de 320 × 320 × 12, y luego después de una operación de convolución de 32 núcleos de convolución, finalmente construye un mapa de características de 320 × 320 × 32 ❖ Cuello: Una serie de capas de red que mezclan y combinan características de imagen y pasan las características de imagen a la capa de predicción. Yolov5 utiliza la estructura FPN-PAN, la estructura CSP2 diseñada por CSPNet y PANET como Neck para agregar características. El cuello se utiliza principalmente para	PyTorch como un paquete de Python que provee dos cualidades: ❖ Cálculos con tensores con aceleración de una Graphic Processing Unit (GPU). Para que se pueda llevar a cabo en GPUs, PyTorch nos permite utilizar CUDA. Basta con saber que CUDA, plataforma de computación paralela, nos sirve para correr nuestros modelos en GPUs. Esto nos permite que nuestra computación científica se ejecute con mayor velocidad a si operáramos en un Central Processing Unit (CPU). ❖ La construcción de redes neuronales construidas sobre un sistema de autogradiente. Este punto nos indica, brevemente, que PyTorch puede realizar, si lo activamos, un seguimiento a los tensores después de pasar por múltiples operaciones, que involucran la creación de una red neuronal, de forma que nos pueda retornar la derivada de la última operación con respecto al tensor en cuestión.

generar pirámides de características, mejorar la detección del modelo de objetos de diferentes escalas y realizar el reconocimiento del mismo objeto de diferentes tamaños y escalas. El extractor de características de la red utiliza una nueva estructura FPN, que mejora la ruta ascendente y mejora la propagación de entidades de bajo nivel. ❖ Cabeza: Puede predecir características de imagen, generar cuadros delimitadores y predecir categorías. La confianza indica la exactitud de la clasificación bajo la condición específica.	Los paquetes en Python pueden ser llamados de 3 formas diferentes dependiendo el requerimiento de trabajo. ❖ Torch: paquete que contiene estructuras de datos para tensores multidimensionales y operaciones matemáticas. ❖ Torch.nn: Creación y entrenamiento de redes neuronales. Los datos ingresados en estos módulos se pasan en forma de tensores. Por ejemplo: al entrenar una "red neuronal convolucional" para imágenes puede usar el módulo "nn.conv2D” ❖ Torch.optim: algoritmos de optimización utilizados para entrenar las redes neuronales. Por ejemplo: algoritmos como SGD, para principiantes, o más avanzados como Adam utilizado para el entrenamiento.

Fuente: (Yao, Qi, & Zhang, 2021, p. 3-4), (Espejel, 2020)
Realizado por: García, Jefferson, 2023.

Teniendo claro cada una de las características que representan la funcionalidad de Yolo v5 frente a Pytorch y el propósito de cada una de estas tecnologías se procede a realizar una escala de Likert bajo especificaciones claras que se necesitan aplicar en base a lo que se requiere desarrollar. Entonces, la idea inicial siempre ha sido poder reconocer objetos y estos objetos deben de ser identificados por sus rasgos y características sobre otros objetos puestos en escena, para ello el prototipo entiende rasgos únicos de cada objeto y lo clasifica según la clase a la que pertenece. Por lo tanto, debe existir una capacidad de reconocimiento de frames por segundo bastante buena, capacidad amplia en cantidad de objetos por cuadro, mientras más objetos pueda reconocer mejor. Para esto las redes neuronales deben de ser entrenadas con gran cantidad de información y en esencia deben de entrenarse en el menor tiempo posible. Se ejecuta entonces una escala presentada en la tabla 1-3b para elegir la tecnología que mejor se acomode al problema que se quiere solucionar.

Tabla 2-3: Escala Likert de actitudes de las tecnologías para la detección de objetos.

	Yolov5	Pytorch
Mayor cantidad de objetos por fotograma	✓	
Computación paralela para GPU		✓
Aprendizaje profundo en el entrenamiento		✓
Facilidad de instalación	✓	
Regresiones de cuadro delimitador	✓	
Mayor número de objetos a reconocer	✓	
Grado de confianza por objeto reconocido	✓	

Fuente: (Yao, Qi, & Zhang, 2021, p. 3-4), (Espejel, 2020), (Pytorch, 2023), (VMWare, 2022), (Analytics, 2023)
Realizado por: García, Jefferson, 2023

Habiendo evaluado cada actitud acorde al problema que se desea resolver se hace una aclaración de cada apartado. Para la cantidad de objetos por fotograma Yolo v5 lleva la ventaja, debido a que Pytorch realiza una clasificación de imágenes de otras imágenes por cada que se le ingrese una, por su lado Yolo v5 dada la condición para la que este entrenada no tiene límite en el número de objetos que pueda reconocer por cada fotograma. Por su parte, Pytorch bajo su compatibilidad con CUDA brinda una computación en GPU que ayuda a la velocidad de procesamiento. La simpleza de su instalación hace que los dos tengan gran acogida por este nivel de valoración. Yolo v5 usa la función de regresión de cuadro delimitador para identificar el objeto de interés a reconocer y adicional a esto gracias a que se ayuda en los pesos de COCO Más de 300000 imágenes de las cuales 220000 están categorizadas. Se sabe que puede llegar a tener 1.5 millones de instancias de objetos, 80 categorías de objetos, 91 clasificaciones de entidades. Como última evaluación Yolo v5 agrega un grado de confianza sobre el cuadro delimitador que encierra el objeto reconocido que nos ayuda a saber que tan seguro es el prototipo de que el objeto a reconocer en realidad es el de interés.

Como análisis sin duda Yolo v5 ofrece determinar una solución a la problemática planteada, pero sería algo ilógico desperdiciar las virtudes de Pytorch sobre la computación paralela en GPU, que ayuda a un aprendizaje más profundo en la última capa de neuronas, dándoles agilidad de aprendizaje en cada proceso de entrenamiento. Por esto que a pesar de elegir a Yolo v5 como la tecnología para la detección de objetos, se harán uso de los beneficios de GPU que brinda Pytorch para una solución más rápida ya que es viable en el uso de las librerías de Python.

Cabe recalcar que Yolo tiene una amplia serie de algoritmos, y en la actualidad a pesar de existir Yolo v6, se elige el algoritmo de Yolo v5 debido a que la versión 6 está dedicada al uso industrial

y al no estar perfectamente establecida se espera la colaboración del público en general para que su código esté libre de errores.

3.9. Determinación de la Zona de Peligro

Dados los estudios de (Ponce, 2017, p. 14) se comprende que la presencia de objetos rotos o con filos cortantes, adornos frágiles, estiletes, tijeras, agujas, entre otros, que generalmente se encuentran al alcance de los niños, podrían desencadenar heridas. Y como adición, los infantes menores de cinco años principalmente los que son muy extrovertidos, vivarachos e intranquilos, (Domingos, Alves & López, 2011, p. 12), son los más propensos a tomar contacto con estos objetos de riesgo. Por esto es por lo que se considera importante determinar una zona de peligro en donde sea más recurrente encontrarse con este tipo de objetos que atentan contra la integridad física de los infantes.

La razón dicta entonces que es más probable encontrar un cuchillo, un tenedor , un vaso de cristal en la cocina, siendo esta área definida como la primera zona de peligro con sus objetos detallados con anterioridad. Por su parte, las habitaciones de los padres adquieren la etiqueta de la segunda zona de peligro, esto debido a que es muy regular encontrarse con tijeras, planchas de ropa y estiletes, que son objetos que del mismo modo representan un peligro latente.

Con esto se quiere evitar que los niños resulten lastimados ya que según (Fiorentino, Molise & Stach, 2014, p.14-16). heridas, quemaduras, cortes, envenenamiento, entre otras, son considerados los responsables de las principales atenciones médicas dentro del nivel de salud primario y secundario.

3.10. Equipos y Materiales.

Para el desarrollo del prototipo de emisión de alertas en tiempo real, se hizo uso de un computador MSI modelo crosshair 15 que facilita el procesamiento de imágenes en su GPU Nvidia, del mismo modo dos celulares, uno de marca Huawei P20 lite y un Xiaomi redmi para la captura de video en tiempo real, un router para la conectividad inalámbrica tanto de los celulares como del computador, a continuación, se explica de manera más detallada cada uno de los materiales.

3.10.1. Computador.

El prototipo necesita procesar gran cantidad de imágenes y datos, que mejor manera de hacerlo en NVIDIA, que gracias a su capacidad brinda procesamiento gráfico, por esto que el computador seleccionado en la ilustración 1-3 cuenta con GPU y sus características se detallan en la tabla 2-3.

Ilustración 1-3: Computador MSI CROSSHAIR 15

Fuente: (MSI Latam, 2023)

Tabla 3-3: Características MSI CROSSHAIR 15

Marca	MSI
Modelo	CROSSHAIR 15
Procesador	11ª Gen. Intel® Core™ i7
Velocidad del Procesador	2.3 GHz (frecuencia turbo máxima 4.6 GHz)
Tarjeta gráfica	GeForce RTX™ 3060 6GB GDDR6
Memoria RAM	16 GB
Almacenamiento	1 TB SSD
Wifi	802.11 a /b/g Wi-Fi 4 (802.11n), Wi-Fi 5 (802.11ac)
Sistema Operativo	Windows 10 Home 64 bits

Fuente: (MSI Latam, 2023)

Realizado por: García, Jefferson, 2023

3.10.2. Dispositivo móvil.

Por ser una parte esencial para ejecutar la visión artificial, se recomienda un teléfono que tenga buena resolución para la toma de videos e imágenes, pues la cámara de este dispositivo hace la vez de la retina en el proceso de visión por computador, por ello se describe el teléfono usado en el proyecto para la monitorear el ingreso del infante a una zona de peligro. Ver la tabla 3-3 y 4-3.

Tabla 4-3: Características de teléfono Xiaomi POCO X4 5G

Dimensión y peso	164,2 x 76,1 x 8,1 milímetros 205 gramos
Pantalla	AMOLED de 6,67 pulgadas Formato 20:9 FullHD+ (2.400 x 1.080 píxeles) Rango de color DCI-P3 Tasa de refresco: 120Hz Muestreo táctil: 360Hz Brillo de 1.200 nits (pico) Gorilla Glass 5
Procesador	Qualcomm Snapdragon 695 5G GPU Adreno 619
Memoria RAM	6 GB LPDDR4x
Almacenamiento interno	128 GB UFS 2.2 Ampliable con tarjetas microSD 1 TB
Cámaras	Trasera de 108 MP f/1.9 Gran angular: 8 MPf/2.2, 118º Macro 2 megapíxeles f/2.4 Delantera de 16 MP f/2.4
Batería	5.000 mAh Carga rápida de 67W

Fuente: (García, 2022)

Realizado por: García, Jefferson, 2023

Tabla 5-3: Características de teléfono Huawei P20 lite

Dimensión y peso	164,2 x 76,1 x 8,1 milímetros 205 gramos
Pantalla	5,84 pulgadas IPS LCD FullHD+ a 2.280 x 1.080 píxeles 432 píxeles por pulgada Aspecto 19:9

Procesador	Kirin 659 a 2,36GHz GPU Mali-T830 MP2
Memoria RAM	4GB
Almacenamiento interno	64GB más microSD
Cámaras	Trasera de 16 megapíxeles f/2.2 2 megapíxeles Flash LED Enfoque PDAF Vídeo 1080p Delantera 16 megapíxeles Vídeo 1080p
Batería	3.000 mAh con carga rápida 9V/2ª

Fuente: (Fernández, 2022)

Realizado por: García, Jefferson, 2023

3.10.3. Router.

Se tiene que establecer una red WLAN, para que los dispositivos que tiene cámara se conecten a la misma red que nuestra PC, así se logra establecer las cámaras de los teléfonos como cámaras inalámbricas web, para detectar los objetos con mejor resolución y en ángulos de ubicación determinados. Para eso se hace uso de un Router inalámbrico N300 D-LINK DIR-615 2.4GHz como se muestra en la ilustración 2-3 y sus características en la tabla 5-3.

Ilustración 2-3: Router D-link

Fuente: (Tecnit, 2023)

Tabla 6-3: Especificaciones Router D-link

Marca	D-Link
Modelo	DIR 615
Velocidad de transmisión	Velocidad inalámbrica de hasta 300 Mbps
Banda	2.4GHz
Antenas	Dispone de 2 antenas omni fijas de 5dBi
Puertos	4 puertos LAN y 1 puerto WAN RJ-45 Fast Ethernet 10/100 Mbps
Modos	Router, Repetidor.

Fuente: (Tecnit, 2023)

Realizado por: García, Jefferson, 2023

3.11. Desarrollo Del Prototipo.

Se trata de una instalación bastante simple, solo se debe tener en cuenta que Python funciona y este habilitado para trabajar bajo diferentes versiones, por lo tanto, cada librería instalada deberá ser compatible con la versión de Python adecuada. De otro modo el preámbulo introducido no hace más que referencia a que si se quiere evitar cualquier tipo de error al instalar los paquetes requeridos ya sea por diferencia entre las versiones o por paquetes existentes por trabajos ajenos al presente, se recomienda hacer uso de un contenedor que permita instalar la versión de python acorde a los requerimientos de Yolo v5.

3.11.1. Instalación de anaconda.

Es bastante simple e intuitivo, solo se tiene que visitar la página oficial de Anaconda, acceder al link de descarga en base a los requerimientos de su sistema de computación y realizar los siguientes pasos.

1. Diríjase a la carpeta que contiene la descarga y ejecute el archivo instalador, es necesario que otorgue los permisos necesarios para evitar problemas.
2. De clic en siguiente.
3. Lea los términos y condiciones antes de aceptar.
4. Se recomienda que instale Just Me, que instalará Anaconda Distribution solo en la cuenta de usuario actual. Seleccione solo una instalación para Todos los usuarios si necesita instalar para todas las cuentas de usuarios en el equipo (lo que requiere privilegios de administrador de Windows).
5. Clic en siguiente.

6. Seleccione una carpeta de destino para instalar Anaconda y haga clic en siguiente.

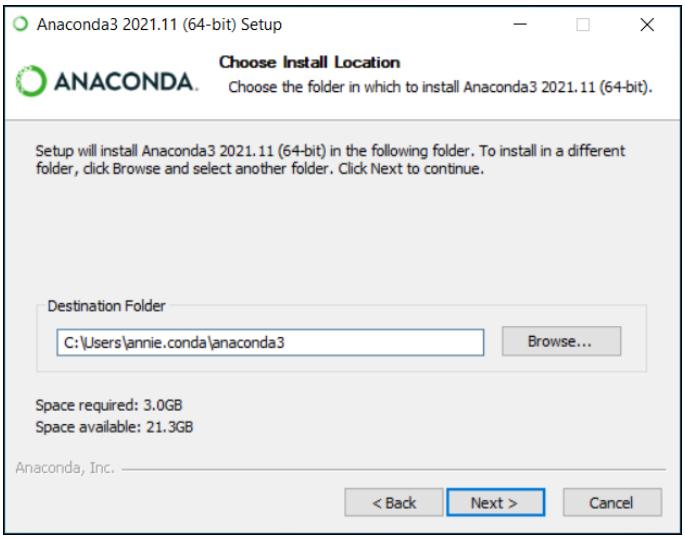


Ilustración 3-3: Selección de la carpeta de destino para la instalación.

Fuente: (Anaconda. Documentación, 2023)

7. Elija agregar Anaconda a su variable de entorno PATH o registrar Anaconda como su Python predeterminado.

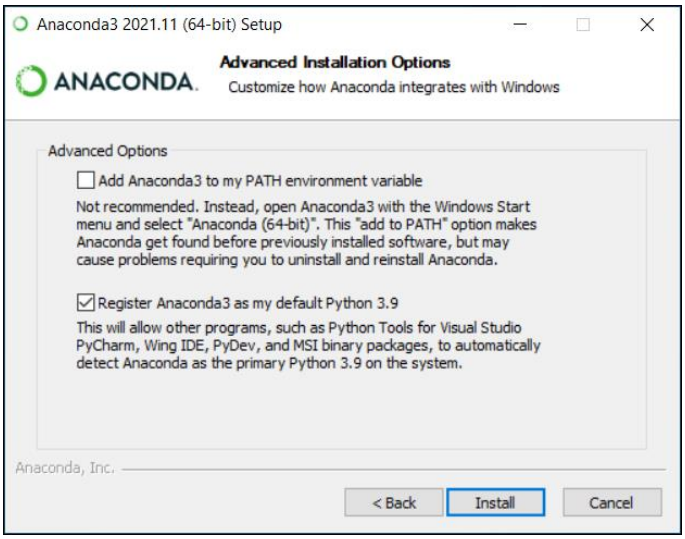


Ilustración 4-3: Elegir la variable de entorno PATH

Fuente: (Anaconda. Documentación, 2023)

8. Haga clic en Instalar. Si desea ver los paquetes que Anaconda está instalando, haga clic en Mostrar detalles.
9. Haga clic en Siguiente.

10. Después de una instalación exitosa, verá el cuadro de diálogo "Gracias por instalar Anaconda", como se muestra en la ilustración 5-3.

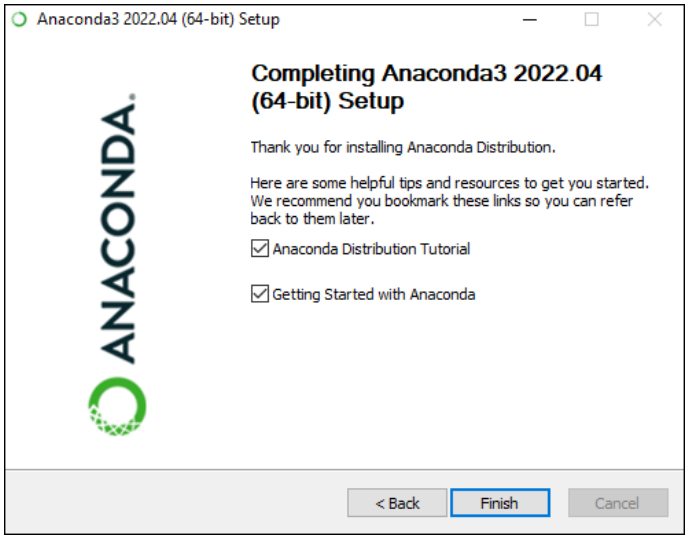


Ilustración 5-3: Finalización.

Fuente: (Anaconda. Documentación, 2023)

Una vez instalado los paquetes necesarios solo resta crear los ambientes de desarrollo para poder ejecutar e instalar las librerías necesarias.

11. En su barra de búsqueda identifique Anaconda Prompt. Ver ilustración 6-3.

Ilustración 6-3: Prompt de anaconda.

Realizado por: García, Jefferson, 2023.

12. En línea de comando se debe ejecutar el siguiente comando: *conda create -n AmbienteTrabajo python=3.8*, AmbienteTrabajo es el nombre del ambiente y python 3.8 es la versión requerida para la compatibilidad de las librerías. Ver ilustración 7-3.

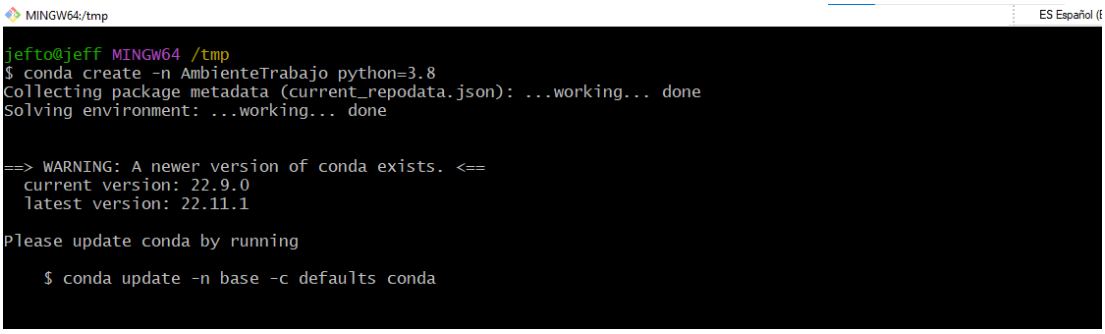


Ilustración 7-3: Creación del Ambiente de trabajo

Realizado por: García, Jefferson, 2023.

13. Teclee yes o y como se muestra en la ilustración 8-3 para que su ambiente se cree de forma correcta.

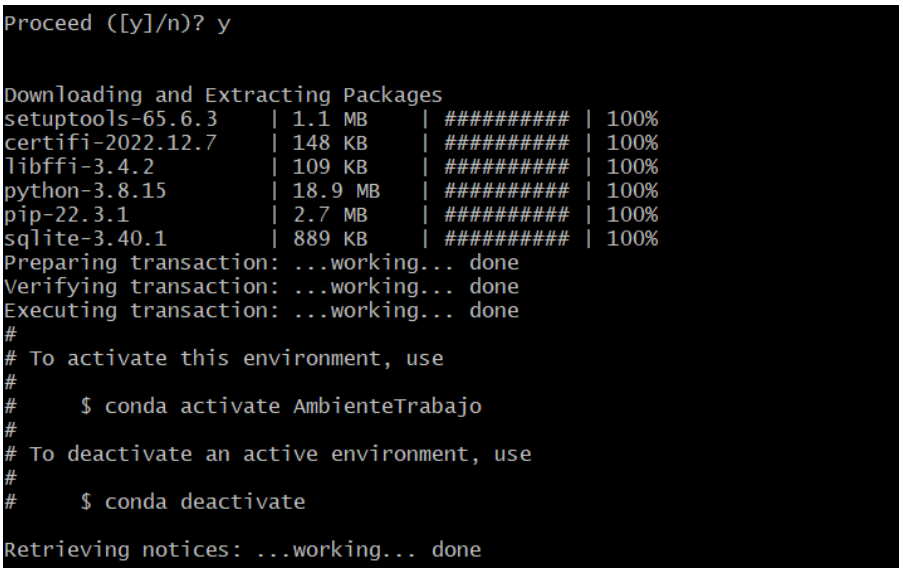


Ilustración 8-3: Ambiente creado

Realizado por: García, Jefferson, 2023.

14. Para activar el ambiente de desarrollo solo ingrese el comando *conda actívate AmbienteTrabajo* o en su defecto *source actívate AmbienteTrabajo.*

3.11.2. Descarga de librerías para el funcionamiento de Yolo v5.

Todo el Proyecto ha sido respaldado en Git Hub su interfaz se muestra en la ilustración 9-3, esto facilita la descarga de todo el contenido para una breve aplicación y demostración del funcionamiento del prototipo, cabe recalcar que todos los paquetes que no han sido personalizados son obtenidos del creador de Yolo v5 en este caso de ULTRALYTICS. Por lo tanto, se puede acceder al link de descarga de Prototipo de emisión de alertas en tiempo real en zonas de peligro para el cuidado de niños de 1 a 5 años (github.com). Mostrando los siguiente:

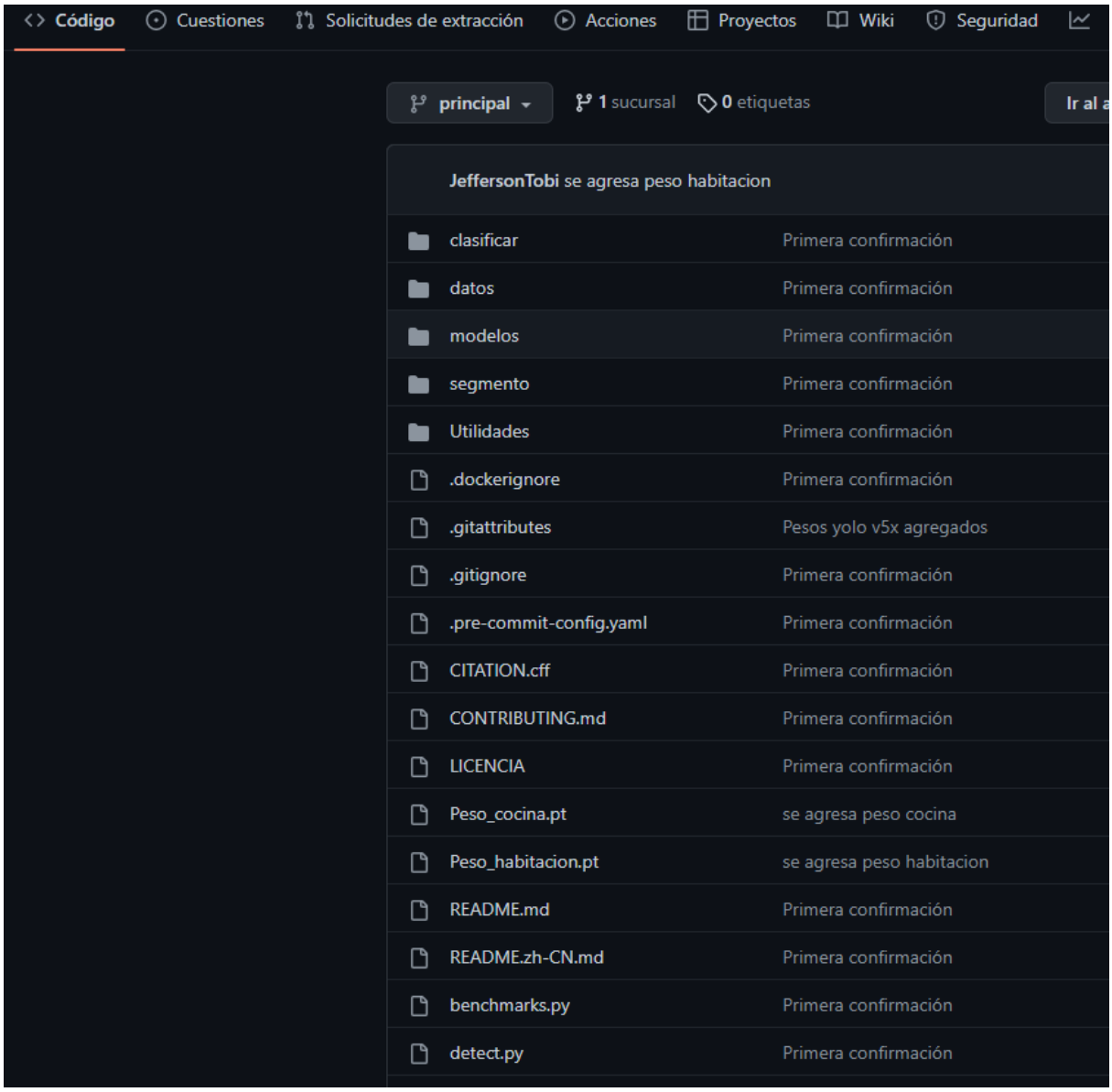


Ilustración 9-3: Documentación respaldada en Git Hub para la facilitar la descarga

Realizado por: García, Jefferson, 2023

Para obtener todo, solo tiene que ejecutar por la línea de comando *git clone https://github.com/JeffersonTGS/Tesis.git.* El enlace se lo puede copiar del apartado código en el recuadro verde.

Hasta este punto, usted debe contar con todos los archivos ejecutables y editables para el funcionamiento del prototipo. Solo resta cambiarse al directorio raíz de la carpeta descargada y continuar el proceso.

3.11.3. Librerías que requieren ser instaladas.

Gitpython, ipython, matplotlib>=3.2.2, numpy>=1.18.5, opencv-python>=4.1.1, Pillow>=7.1.2, Psutil, PyYAML>=5.3.1, requests>=2.23.0, scipy>=1.4.1, thop>=0.1.1, torch>=1.7.0, torchvision>=0.8.1, tqdm>=4.64.0.

Para simplificar el proceso, todas las librerías se encuentran en un archivo.txt denominado requirements.txt, esto simplifica la instalación unitaria y demorada de cada librería. Ejecute entonces *pip install -r requirements.tx.* Nota: no olvide que tiene que hacerlo dentro del directorio raíz y con su ambiente de trabajo de anaconda activado.

3.11.4. Verificación de los documentos.

Se le facilita archivos ejecutables para realizar una prueba de ejecución con pesos propios de Yolo V5 con rasgos e identificaciones de la dataset de COCO. Lo único que se tiene que efectuar es la línea de comando *python detect.py --weights yolov5s.pt --img 640 --conf 0.25 --source data/images.*

Yolo realizará una detección sobre el directorio images en cada una de las imágenes que contiene esa carpeta, etiquetando objetos que constan en los pesos de yolv5s. Revisar el directorio runs/detect/exp. Si lo que desea es realizarlo sobre video o cámara web, lo único que tiene que hacerse en cambiar el directorio de búsqueda para la detección, en este caso, *source video.mp4* o *source 0,1,2,etc.* Respectivamente.

3.11.5. Modelo personalizado.

La dataset de COCO nos brinda 80 clases de objetos, cada uno con su respectiva etiqueta, es por esto que para poder identificar un objeto que difiere de estas 80 clases se realiza una recolección de imágenes muy amplia, cada imagen se archiva conjuntamente con su etiqueta par que Yolo v5 pueda empezar el entrenamiento.

3.11.5.1. Creación de la metada.

La recolección requiere de tiempo puesto que se realiza un filtrado de imágenes con extensión .jpg que contengan el objeto de interés. De que tan variadas sean las imágenes y que las mismas

tengan diferentes grados de luminosidad dependerá la calidad del entrenamiento, mientras más variabilidad exista, más robustos serán los pesos de la red neuronal entrenada.

3.11.5.2. Etiquetado de las imágenes.

Este proceso es indispensable, de esto depende que la red neuronal aprenda a identificar una clase de la otra. Existe varias formas de añadir la etiqueta correspondiente a cada imagen y para el presente proyecto se usó la herramienta en línea **makesense.ai** (ilustración 10-3), aquí, se puede crear el número de clases pertinentes al trabajo que se va a realizar, luego subir las imágenes y empezar a etiquetar en el orden que le corresponda a cada clase.

Ilustración 10-3: Etiquetado de las imágenes según la clase que corresponda

Fuente: (Makesene,2023)

Realizado por: García, Jefferson, 2023.

3.11.5.3. Orden de las imágenes y etiquetas.

Una vez obtenida la dataset para el modelo personalizado, Yolo necesita una ruta clara para establecer el camino de donde tiene que tomar los datos correspondientes para cada clase y la etiqueta que le corresponde .

Por esto, se debe crear una carpeta llamada *data* y dentro de esta, dos carpetas con nombres de *images* y *labels.* A su vez, cada carpeta debe contener dos carpetas que se identifiquen con *train* y *val.* Entonces, la carpeta *train* dentro de *images* debe contener las imágenes que van a ser usadas para entrenar el modelo, del mismo modo la carpeta *val* debe contener un porcentaje menor de imágenes que sirven de validación del modelo una vez terminado el proceso de

entrenamiento. Queda claro entonces que del mismo modo para la carpeta *train* y *val* dentro de *labels* debe contener las etiquetas de las imágenes que son objeto de entrenamiento y de validación respectivamente.

3.11.6. Entrenamiento del modelo personalizado.

Es tan simple como ejecutar una línea de comando, teniendo en cuenta que cada archivo a correr debe estar contenido dentro del directorio raíz. Siendo así, solo se debe crear un archivo con extensión .yaml como el mostrado en la ilustración 11-3, este archivo debe contener la dirección de la carpeta *data/images/train* del mismo modo que *data/images/val* y las clases en el orden de la etiqueta proporcionada.

```yaml
train: /Users/jefto/OneDrive/Escritorio/yolo5/data/images/train  # train images (relative to 'path') 128 images
val: /Users/jefto/OneDrive/Escritorio/yolo5/data/images/val # val images (relative to 'path') 128 images
#test:  # test images (optional)

# Classes
names:
  0: cuchillo
  1: vaso
  2: Hijo(a)
  3: tenedor
```

Ilustración 11-3: Configuración del archivo.yaml

Realizado por: García, Jefferson, 2023

Una vez creado el archivo.yaml se tiene que ejecutar el siguiente comando para que el modelo empiece a entrenar con los datos personalizados. *python train.py --img 640 --batch 16 --epochs 150 --data {dataset.location}/data.yaml --weights yolov5x.pt --cache*

3.11.7. Correr modelo sobre GPU.

Es posible que todo el proceso hasta aquí se haya ejecutado en su CPU, pero si cuenta con una tarjeta gráfica es factible que se pueda correr en GPU y es aquí donde Pytorch brinda el uso de tensores que ejecutan CUDA para el procesamiento gráfico en tarjetas NVIDIA e incluso brindando un aprendizaje más rápido en el proceso de entrenamiento procesando los gráficos en GPU.

3.11.7.1. Instalación de PyTorch.

Este paso depende del tipo de tarjeta gráfica que se tenga, en el presente trabajo se usa una GeForce RTX™ 3060 6GB GDDR6 y la versión de CUDA 11.6, solo resta elegir el sistema que se está ocupando, para una muestra se ejemplifica en la ilustración 12-3.

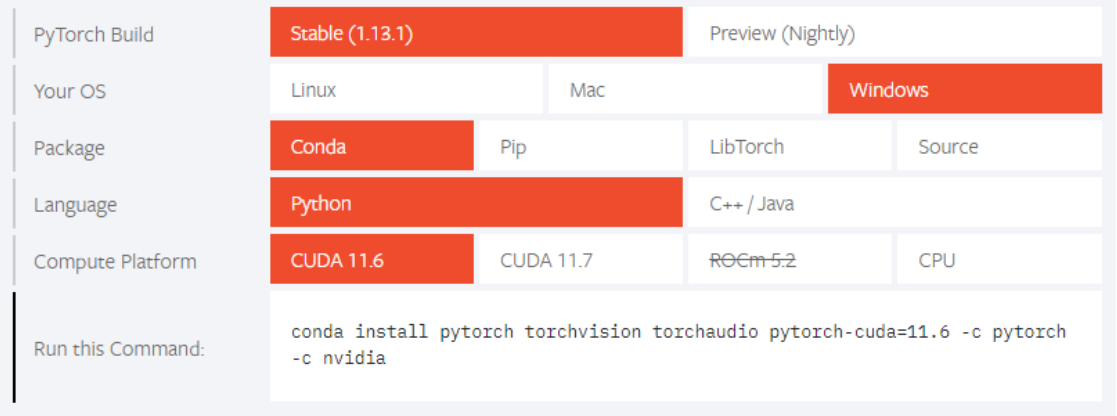


Ilustración 12-3: Instalación de Pytorch.

Fuente: (PyTorch ORG, 2023)

Terminado este proceso, los modelos ejecutados tanto para entrenamiento como para detección están corriendo en GPU (ilustración 13-3).

```
YOLOv5  v6.2-230-g1079ca82 Python-3.8.13 torch-1.12.1 CUDA:0 (NVIDIA GeForce RTX 3060 Laptop GPU, 6144MiB)
```

Ilustración 13-3: Yolov5 ejecutado sobre GPU.

Realizado por: García, Jefferson, 2023.

3.11.8. Implementación para el proceso de detección.

En este punto se cuenta con cada uno de los requerimientos instalados en el ordenador de manera que se puede ejecutar la detección en tiempo real por ello que a continuación se detallan las librerías que de forma obligatoria deben de ser instaladas dependiendo de la versión de python que se use y se explica a demás cada proceso para poder obtener la detección de objetos y como aplicarla para que funcione sobre cualquier sistema.

3.11.8.1. Importación de librerías instaladas

Instalados todos los requerimientos del sistema es necesario importar cada una de las librerías para la correcta ejecución de cada fragmento de código, cada ejecución necesita de su módulo y de su paquetería correspondiente. Ver la ilustración 14-3.

```
import argparse
import os
import platform
import sys
from pathlib import Path
import torch
import math
#import urllib3
import requests
import json
from urllib.request import urlopen
FILE = Path(__file__).resolve()
ROOT = FILE.parents[0]  # YOLOv5 root directory
if str(ROOT) not in sys.path:
    sys.path.append(str(ROOT))  # add ROOT to PATH
ROOT = Path(os.path.relpath(ROOT, Path.cwd()))  # relative
from models.common import DetectMultiBackend
from utils.dataloaders import IMG_FORMATS, VID_FORMATS, LoadImages, LoadScreenshots, LoadStreams
from utils.general import (LOGGER, Profile, check_file, check_img_size, check_imshow, check_requirements, colorstr, cv2,
                           increment_path, non_max_suppression, print_args, scale_boxes, strip_optimizer, xyxy2xywh)
from utils.plots import Annotator, colors, save_one_box
from utils.torch_utils import select_device, smart_inference_mode
```

Ilustración 14-3: Librerías importadas.

Realizado por: García, Jefferson, 2023

3.11.8.2. Selección de la cámara.

En el archivo ejecutable hay un apartado de *source* que indica el número de cámara que se hace uso para la captación de video en tiempo real, el número de cámara dependerá de la zona en la que se necesite la captación de objetos. Por tanto, para no ejecutar un proceso complejo para la obtención de las cámaras web de alta definición, se recomienda utilizar Irun Webcam que permite acceder a las cámaras celulares de alta resolución de forma inalámbrica para la captación de video como si fuera una cámara web común. Su interfaz es la que se puede apreciar en la ilustración 15-3.

Ilustración 15-3: Selección de la cámara mediante Iriun

Fuente: (Irun, 2020).

Realizado por: García, Jefferson, 2023

Tan solo queda elegir el número de cámara pertinente en el código como se muestra en la ilustración 16-3.

```
parser.add_argument('--source', type=str, default=ROOT / '2', help='file/dir/URL/glob/screen/0(webcam)')
```

Ilustración 16-3: Ejemplo de cómo elegir la cámara, en este caso es la opción 2.

Realizado por: García, Jefferson, 2023.

3.11.8.3. Proceso de captación de imagen.

Por su parte el proceso de detección en el código por webcam toma un valor mayor a 1 si se ejecuta por webcam y guarda en las respectivas variables. Ver el código de la ilustración 17-3.

```
for i, det in enumerate(pred):  # per image

    seen += 1
    if webcam:  # batch_size >= 1
        p, im0, frame = path[i], im0s[i].copy(), dataset.count


    else:
        p, im0, frame = path, im0s.copy(), getattr(dataset, 'frame', 0)

    p = Path(p)  # to Path
    save_path = str(save_dir / p.name)  # im.jpg
    txt_path = str(save_dir / 'labels' / p.stem) + ('' if dataset.mode == 'image' else f'_{frame}')  # im.txt

    gn = torch.tensor(im0.shape)[[1, 0, 1, 0]]  # normalization gain whwh
    imc = im0.copy() if save_crop else im0  # for save_crop
    annotator = Annotator(im0, line_width=line_thickness, example=str(names))
    if len(det):
```

Ilustración 17-3: Proceso de captación por webcam

Realizado por: García, Jefferson, 2023.

Por otro lado cv2 estructura los datos procesando y analizando las imágenes realizando un análisis estructural para indicar que el objeto esta correctamente situado, por eso se muestra la estructura del código en la ilustración 18-3.

```
if view_img:
    if platform.system() == 'Linux' and p not in windows:
        windows.append(p)
        cv2.namedWindow(str(p), cv2.WINDOW_NORMAL | cv2.WINDOW_KEEPRATIO)  # allow window resize (Linux)
        cv2.resizeWindow(str(p), im0.shape[1], im0.shape[0])

    cv2.imshow('ZONA 1', im0)

    cv2.waitKey(1)  # 1 millisecond
```

Ilustración 18-3: Indicador de objetos correctamente situado por Cv2

Realizado por: García, Jefferson, 2023.

3.11.8.4. Cálculo de la distancia entre el infante y el objeto de peligro.

Como se aprecia en la ilustración 19-3, primero se registran las clases que se desea detectar para luego poder comparar la clase encontrada con la clase guardada en el contador de objetos.

```
if names[int(c)] == 'Hijo(a)':

    coohx, coohy = round((c1[0] + c2[0]) / 2), round((c1[1] + c2[1]) / 2)

if names[int(c)] == 'vaso':

    coovx, coovy = round((c1[0] + c2[0]) / 2), round((c1[1] + c2[1]) / 2)

if names[int(c)] == 'cuchillo':

    coocux, coocuy = round((c1[0] + c2[0]) / 2), round((c1[1] + c2[1]) / 2)
```

Ilustración 19-3: Comparación del objeto detectado con el objeto de interés

Realizado por: García, Jefferson, 2023.

Teniendo claro e identificado el objeto se establece cual es el objeto de peligro para la integridad física del infante y se extrae la distancia entre estas dos clases tal como se muestra en la ilustración 20-3.

```
f coohx_ant is not None and coohy_ant is not None and coovx_ant is not None and coovy_ant is not None ar

  #Calcular la distancia euclidiana entre los objetos actuales y anteriores
  distancia = ((coohx - coovx_ant) ** 2 + (coohy - coovy_ant) ** 2) ** 0.5

  # Imprimir la distancia en el frame actual
  cv2.putText(im0, "Distancia: {:.2f}".format(round(distancia)), (50, 50), cv2.FONT_HERSHEY_SIMPLEX, 1,
              (0, 255, 0), 2, cv2.LINE_AA)
  dd = f'la distancia del niño al vaso es: {round(distancia)}'
  print(dd)
```

Ilustración 20-3: Cálculo de la distancia entre el infante y el objeto de peligro

Realizado por: García, Jefferson, 2023.

3.11.8.5. Emisión de alertas.

Una vez creado el Bot por Telegram, solo se necesita tener claro su ID mostrado en la ilustración 21-3, este se encargará de enviar las alertas de forma textual tal como se aprecia en la ilustración 22-3.

```
# INICIO DE LA CREACIÓN DEL BOT EN TELEGRAM

r = requests.post('https://api.telegram.org/bot5490359608:AAFANYUd_rg2PlTmCG0w-n3qgarP5pn0XkA/sendMessage',
                  data={'chat_id': '-791568642', 'text': 'su hijo esta cerca de un cuchillo'})
# data = json.loads(r.text)
```

Ilustración 21-3: API para Telegram

Realizado por: García, Jefferson, 2023.

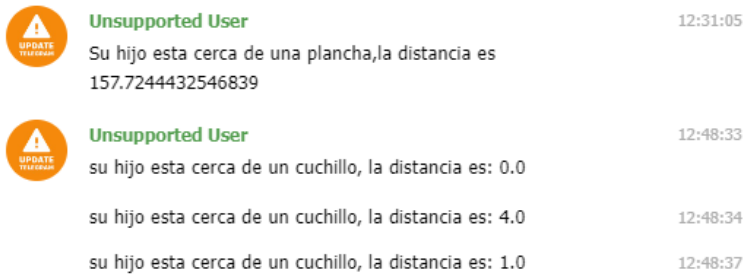


Ilustración 22-3: Alertas en Telegram

Realizado por: García, Jefferson, 2023.

CAPITULO IV

4. MARCO DE ANALISIS E INTERPRETACION DE RESULTADOS.

Aquí se interpreta cada resultado obtenido en los escenarios de pruebas a los que el sistema de detección de objetos fue sujeto, entre los parámetros a evaluar se encuentra la efectividad del modelo al detectar los objetos propuestos y el tiempo de retraso con el que las alertas son emitidas.

4.1. Resultados.

4.1.1. Resultados del entrenamiento personalizado.

El entrenamiento personalizado se lo hace con el objetivo de que el prototipo reconozca solo objeto de interés, con esto se logra reducir el costo computacional que requiere la visión artificial. Además, se pueden reconocer algunos objetos que no vienen en el preentrenamiento de Yolo v5 en su dataset de COCO. Para la primera zona, o zona de la cocina se recolectó una dataset de 4115 imágenes que varían entre cuchillos, tenedores, vasos de cristal y niños de 1 a 5 años. Y para la segunda zona o zona de la habitación se la hizo con un total de 4046 imágenes. El proceso para las dos zonas se lo hizo por un total de 300 épocas de 2 lotes.

4.1.2. Entrenamiento de Yolo v5 sin Pytorch.

Es un proceso tardío, que requiere de mucho tiempo para procesar cada imagen, se logra obtener el 1% de una época de las 300 en 7 minutos, representa un verdadero reto entrenar redes neuronales sobre CPU debido al tiempo que este proceso necesita, para constancia de ello se adjunta la ilustración 1-4.

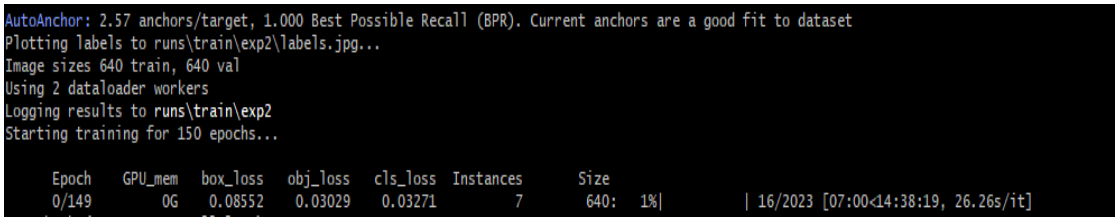


Ilustración 1-4: Entrenamiento de YoloV5 sin Pytorch.

Realizado por: García, Jefferson, 2023.

4.1.3. Entrenamiento de Yolo v5 con Pytorch.

Ventajosamente python cuenta con muchas librerías que facilitan todos los procesos de entrenamiento, una de ellas es PyTorch para el entrenamiento de redes neuronales de forma profunda, para ejecución sobre GPU. La ilustración 2-4 deja en evidencia la rapidez de este proceso dando como resultado el 4% de una época en menos de 54 segundos.

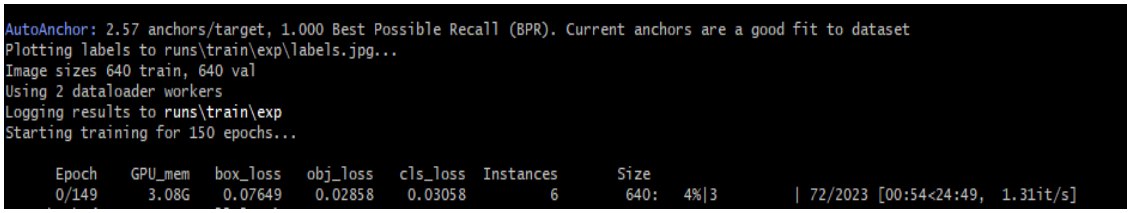


Ilustración 2-4: Entrenamiento de YoloV5 con Pytorch.

Realizado por: García, Jefferson, 2023.

4.1.4. Resultado del entrenamiento de Yolo v5 con dataset personalizada frente a Yolo v5 preentrenado.

Es prudente aclara que la zona 2 de este proyecto no guarda relación con objetos que por defecto Yolov5 ya reconoce por efectos de preentrenamiento, pero si se puede hacer una comparativa de la zona 1 que guarda relación en objetos como cuchillos, vasos de cristal, personas y tenedores, siendo estos una variable en común entre Yolo v5 preentrenado y Yolo v5 personalizado. La curva de función de perdida de entrenamiento y el valor de precisión promedio son dos aspectos por los cuales se va a juzgar cada modelo.

4.1.4.1. Función de pérdida de entrenamiento y Valor de precisión promedio mAP.

La ilustración 3-4 contine la información de la función de pérdida de entrenamiento propio de Yolo v5 para las 80 clases para la cual está entrenada, mientras que la ilustración 4-4 muestra la función de perdida de Yolo v5 personalizado para que reconozca, niños, vaso de cristal, tenedores y cuchillos, mismos objetos que Yolo v5 contiene dentro de su conjunto de datos entrenado.

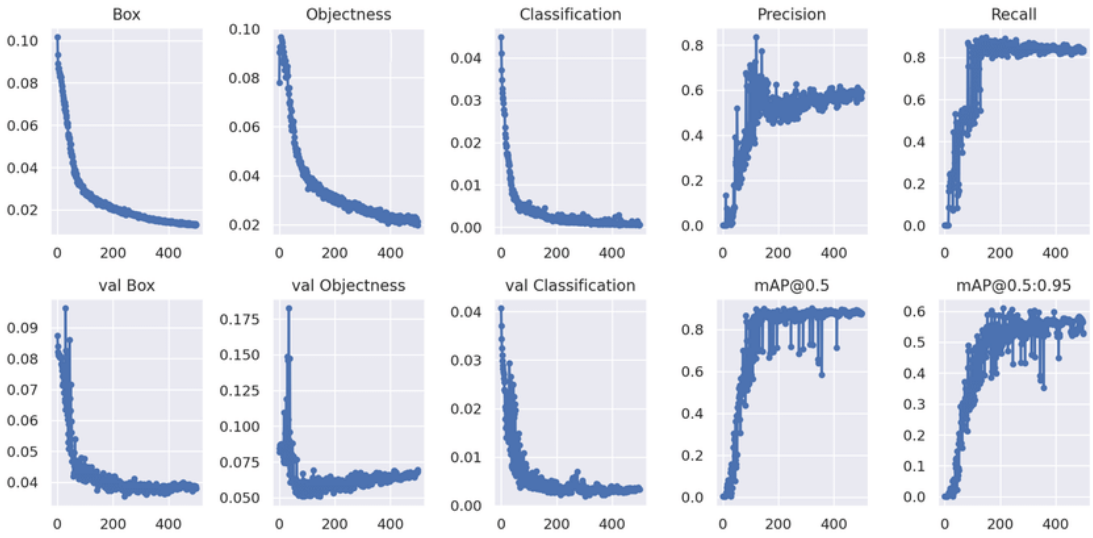


Ilustración 3-4: Función de pérdida de Yolo v5 original y su valor de precisión mAP correspondiente.

Fuente: (Redmon, 2016)

Realizado por: García, Jefferson, 2023.

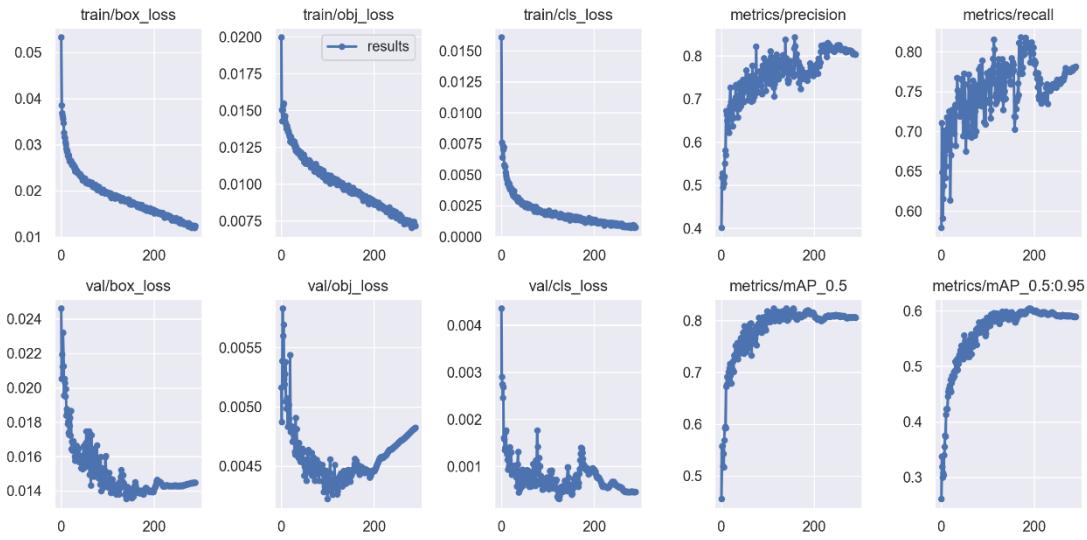


Ilustración 4-4: Función de pérdida de Yolo v5 personalizado y su valor de precisión promedio mAP correspondiente.

Realizado por: García, Jefferson, 2023.

Yolo v5 cuenta con un entrenamiento superior a las 300 épocas, mientras que Yolo v5 personalizado solo cuenta hasta las 300 épocas de entrenamiento. Sin embargo, eso no impide llevar una buena comparación ya que para los dos casos el rango de pérdida se aproxima a cero a partir de la época 200, a partir de donde se mantiene constante.

Yolo v5 en base al valor de precisión promedio mAP en el rango mínimo de 0.5, cumple con el rendimiento con un valor final para todas las clases de 0.817, esto se puede ver de mejor manera en la ilustración 5-4 correspondiente a la zona 1 y para la zona 2 un mAP 0.924 que se muestra en la ilustración 6-4. El mAP se utiliza para medir la calidad del modelo de detección de objetos. Cuanto mayor sea el valor, mayor será la efectividad de detección promedio y mejor será el rendimiento.

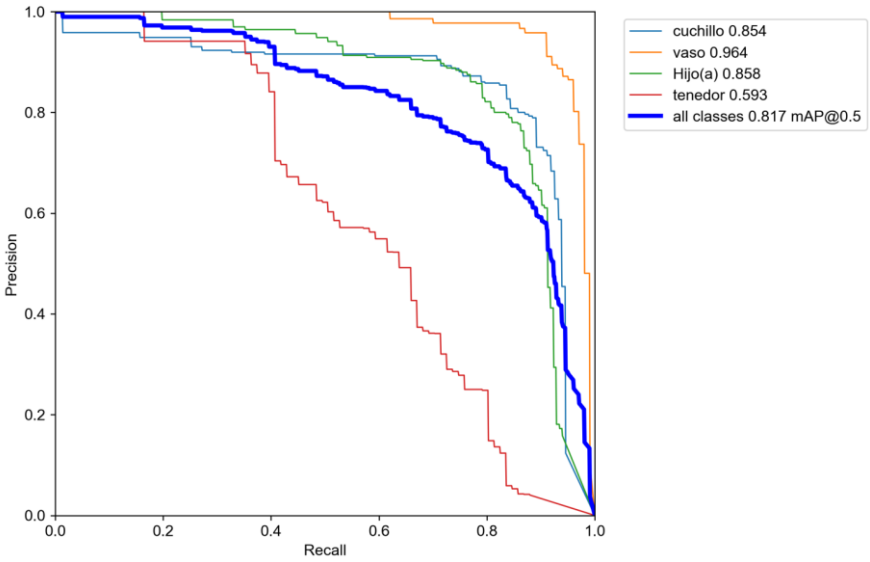


Ilustración 5-4: Valor de precisión promedio mAP correspondiente a la zona 1.

Realizado por: García, Jefferson, 2023.

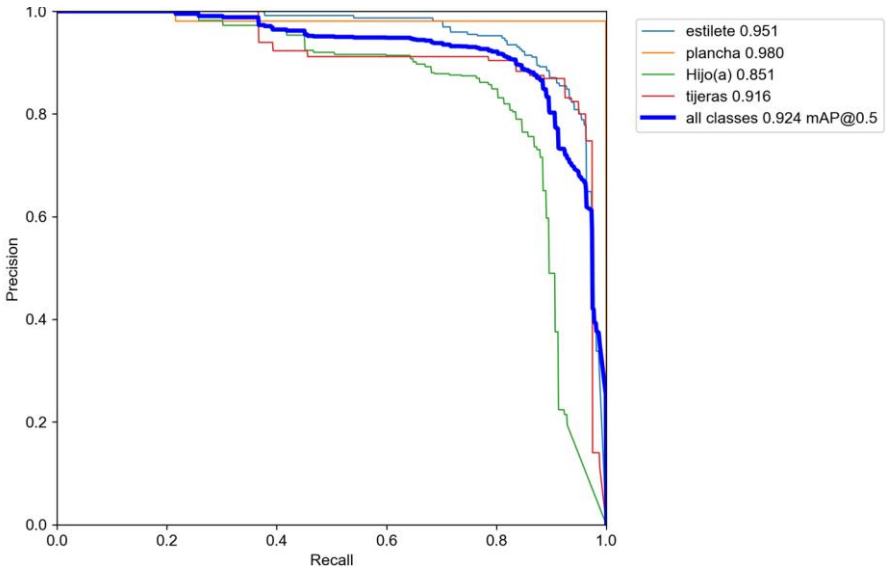


Ilustración 6-4: Valor de precisión promedio mAP correspondiente a la zona 2.

Realizado por: García, Jefferson, 2023.

4.2. Evaluación Del Prototipo.

La evaluación del proyecto se mide en que tan funcionales y verídicos son los datos que arroja el prototipo puesto en escena en un entorno real, evaluando la eficiencia de la alerta temprana y la detección de los objetos en tiempo real.

4.2.1. Matriz de confusión.

Yolo cuenta con una herramienta muy útil basada en el porcentaje de predicciones correctas para las que el modelo fue entrenado, esta herramienta llamada matriz de confusión es creada por Yolo para validar las clases para las que fue entrenado, permite evaluar la exactitud de una clasificación calculando la cantidad de resultados reales reconocidos correctamente con la predicción (verdaderos positivos para resumir VP), los valores reales negativos que fueron predichos como verdaderos (falsos positivos o FP), los valores reales negativos que fueron predichos correctamente como negativos (verdaderos negativos o VN) y por último, los valores reales positivos que fueron predichos como negativos o FN).

La matriz de confusión presentada en la ilustración 7-4, que muestra el grado de certeza del modelo sobre la zona 1 o cocina. Se evalúa el prototipo para la clase cuchillo con 100 imágenes de cuchillos, a partir de esta detección se cuentan los verdaderos positivos y se divide entre la suma de los verdaderos positivos y los falsos positivos $Precisión = \frac{VP}{VP+FP}$. La cantidad de los VP es de 72 esto indica que el modelo detecto 72 cuchillos de 100 y la cantidad de los FP es de 14, por tanto, el modelo predijo cuchillos 14 veces de 100 en donde no existían cuchillos y luego de aplicar la fórmula se tiene un grado de confianza de 0.837. Por contraparte si se calcula el grado de error cuando el modelo detecto chuchillos en donde no habían, se tiene un 0.1627. Estos valores son aproximados a la matriz de confusión emitida por Yolo v5 por lo tanto dada la similitud se establece el correcto funcionamiento de selectividad, la ilustración 8-4 en modo de collage muestra la detección sobre las 100 imágenes proporcionadas en forma de resumen.

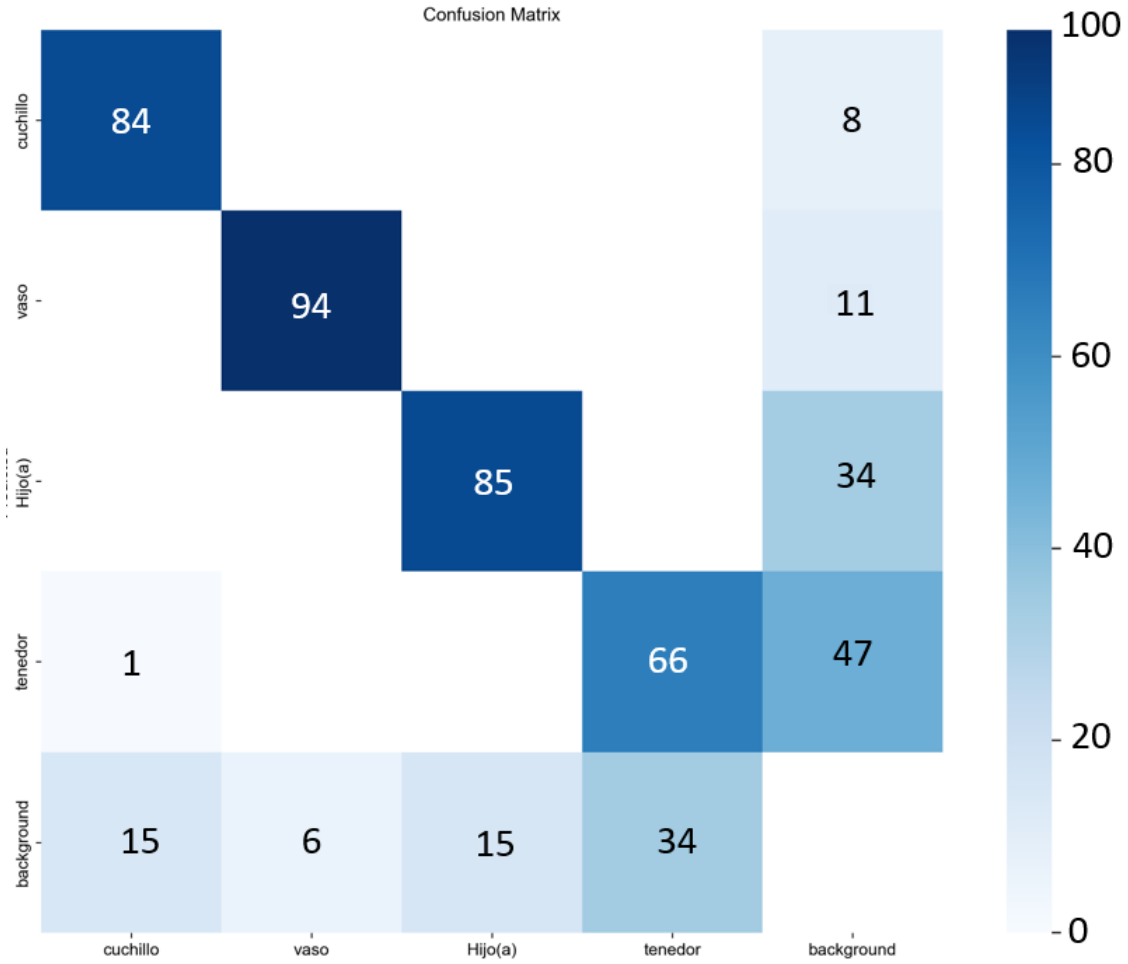


Ilustración 7-4: Matriz de confusión de la zona 1 o cocina.

Realizado por: García, Jefferson, 2023.

La matriz de confusión de la zona 1 o cocina presentada con anterioridad en la ilustración 7-4 da buenos resultados, el valor más alto se obtuvo para la clase vaso con un valor de 94%, seguido de la clase Hijo(a) con un valor de 85% y 84% para la clase cuchillo, por último, la clase tenedor tiene una precisión de 66%.

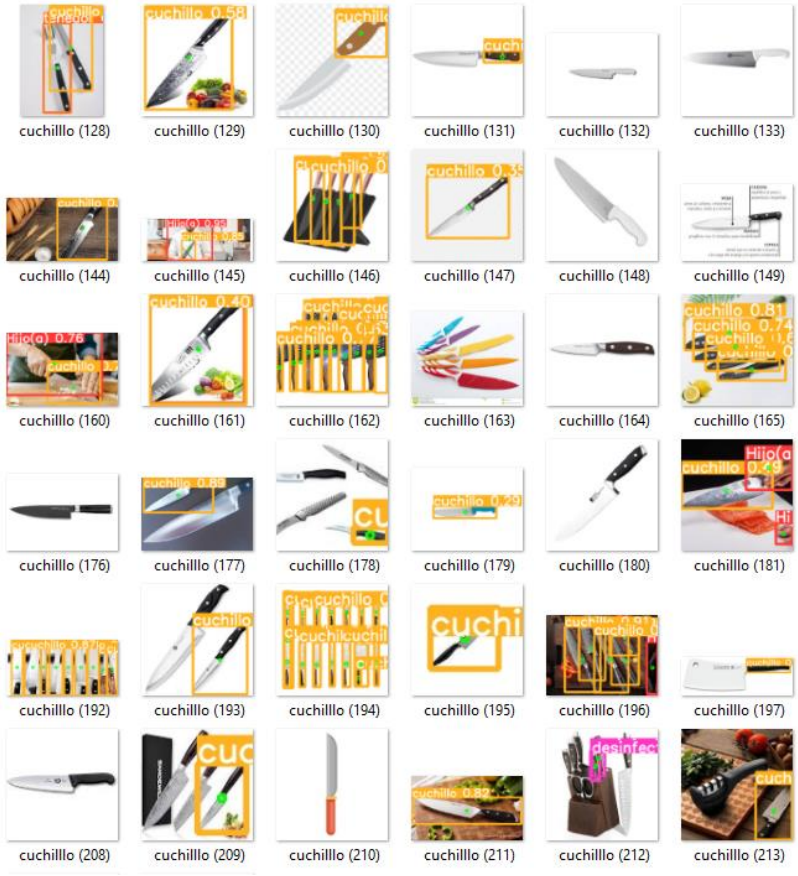


Ilustración 8-4: Sustento de la matriz de confusión

Realizado por: García, Jefferson, 2023.

Se presenta la matriz de confusión para la zona 2 en la ilustración 9-4 con los porcentajes de precisión de cada clase. Aquí los resultados son mucho mejores, incluso la clase plancha alcanza el 100% de precisión en base a la sensibilidad, la clase tijera tiene un valor de 92% que es bastante aceptable 86% y 8%2 para estilete e hijo(a), respectivamente.

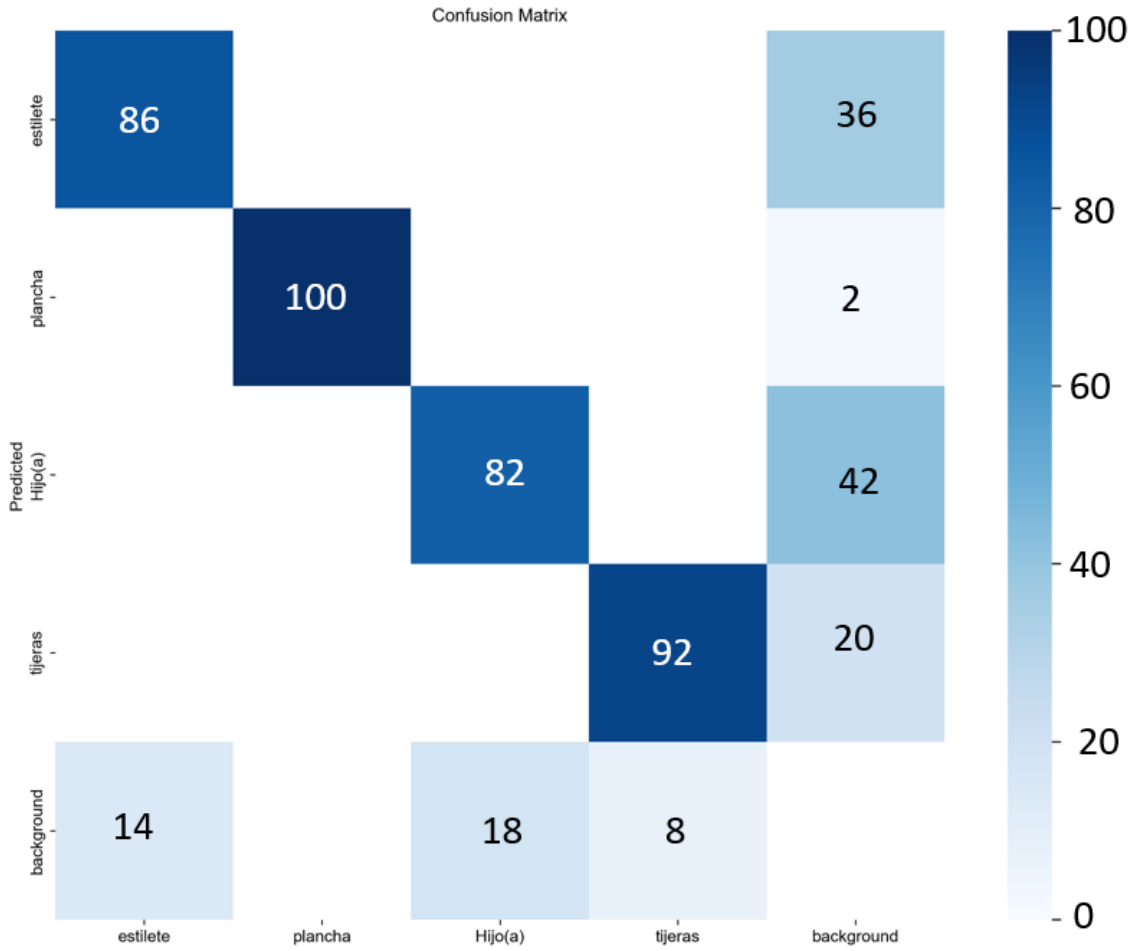


Ilustración 9-4: Matriz de confusión de la zona 2 o habitación

Realizado por: García, Jefferson, 2023.

4.2.2. Evaluación del prototipo sobre un escenario real.

La evaluación del prototipo se la realizó en un escenario real con un niño, los resultados obtenidos son los que se muestran en cada sección a continuación.

4.2.2.1. Yolo v5.

En la ilustración 10-4 a 14-4 se muestran a manera de ejemplo la detección que realiza el prototipo en la zona cocina, en donde el objeto peligroso tomado en cuenta para el ejemplo es un cuchillo. Mediante la experimentación se llega a colocar el móvil a una altura de 2 metros que enfoca al y al objeto de peligro y al niño durante todo el tiempo. Se aprecia que el infante está jugando, pero en un momento de distracción de su tutor, el niño se acerca al cuchillo, en este momento el prototipo entra en acción censando la proximidad que existe entre el niño y el cuchillo, y si en algún momento la distancia entre los dos es menor a 1 metro el sistema alerta

al tutor encargado. Adicionalmente, se aprecia que la alerta sonora incluso pone sobre aviso al mismo infante, y provoca que se aleje del objeto de forma inmediata.

La distancia y la altura de la cámara juegan un papel muy importante para determinar la proximidad entre el niño y el objeto de peligro, las pruebas realizadas con una ubicación de la cámara a 2 metros de altura y distancia hacen que la alerta se active cuando el niño guarda una relación menor a 300 píxeles que equivalen a 1m de distancia entre el objeto de peligro. En perspectiva si la distancia y altura de la cámara aumenta a un claro ejemplo de 3m y se conserva la relación de 300 píxeles la distancia entre el niño y el objeto aumenta a 1,32m. Se debe tener en cuenta que alejar la cámara hace que la detección de los objetos pequeños sea imprecisa, y por esta razón las pruebas realizadas mantienen la relación de 2m.

En la ilustración 10-4 se evidencia al niño jugando y se queda solo frente a un objeto que representa un peligro para su integridad física. Por su parte Yolo v5 se encarga de administrar la etiqueta correspondiente a los dos objetos de interés, junto con el grado de confianza de que tan seguro esta de que creer detectar lo que dice detectar. Este porcentaje de certeza se actualiza en tiempo real para los dos objetos.

Ilustración 10-4: Escenario real, zona de peligro cocina, objeto chuchillo.

Realizado por: García, Jefferson, 2023.

En la ilustración 11-4 el infante se dirige hacia el objeto de peligro y Yolo v5 se encarga de censar siempre su posición y la posición de cuchillo para encontrar la distancia entre el infante y el objeto.

Ilustración 11-4: Niño avanzando hacia el objeto de peligro.

Realizado por: García, Jefferson, 2023.

Se puede evidenciar en la ilustración 12-4 que la distancia entre el niño y el objeto se redujo, por tanto, en este paso Telegram recibe las alertar pertinentes y la alarma sonora entra en funcionamiento.

Ilustración 12-4: Distancia reducida, inicio de la emisión de alerta.

Realizado por: García, Jefferson, 2023.

En la ilustración 13-4 la distancia entre el infante y el cuchillo es evidentemente comprometedora, pero el niño es asustado por la alerta sonora que llama su atención, si bien es cierto que la alerta sonora está destinada para que el tutor a cargo se percate de manera inmediata

del suceso, el infante se aleja del objeto sin la necesidad de que su tutor intervenga como se puede evidenciar en la ilustración 14-4.

Ilustración 13-4: Distancia reducida, inicio de la emisión de alerta.

Realizado por: García, Jefferson, 2023.

Ilustración 14-4: Infante alejándose del objeto de peligro asustado por la alerta.

Realizado por: García, Jefferson, 2023.

4.2.2.2. Yolo v5 Personalizado.

Una vez probado Yolo v5 se evidencia que algunas clases de interés no se encuentran dentro de sus clases pre entrenadas como es el caso de la clase plancha, para ello se efectuó el entrenamiento de la red neuronal de forma que Yolo sea capaz de detectar este objeto en conjunto con la clase Hijo(a).

De la ilustración 15-4 a la ilustración 17-4, se muestra la efectividad del modelo sobre la zona de peligro denominada habitación, evitando que el niño se acerca demasiado y se queme con la plancha.

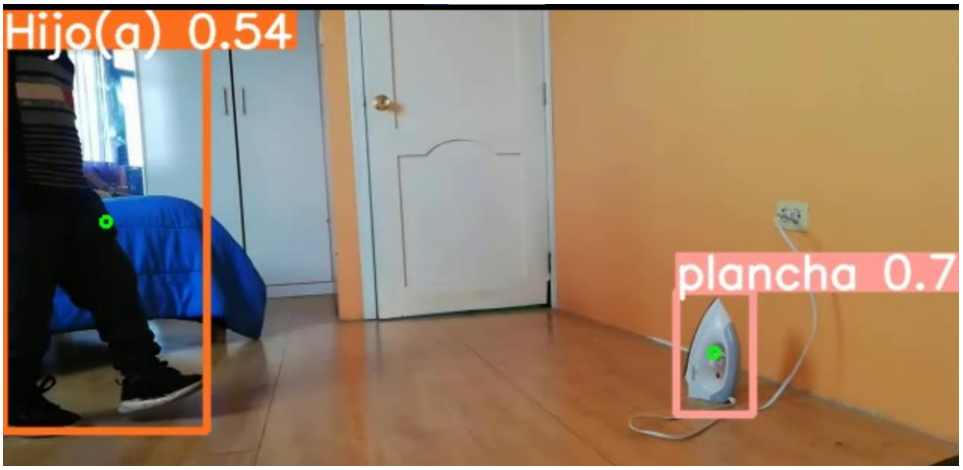


Ilustración 15-4: Escenario real habitación. Clase Hijo(a), plancha

Realizado por: García, Jefferson, 2023.

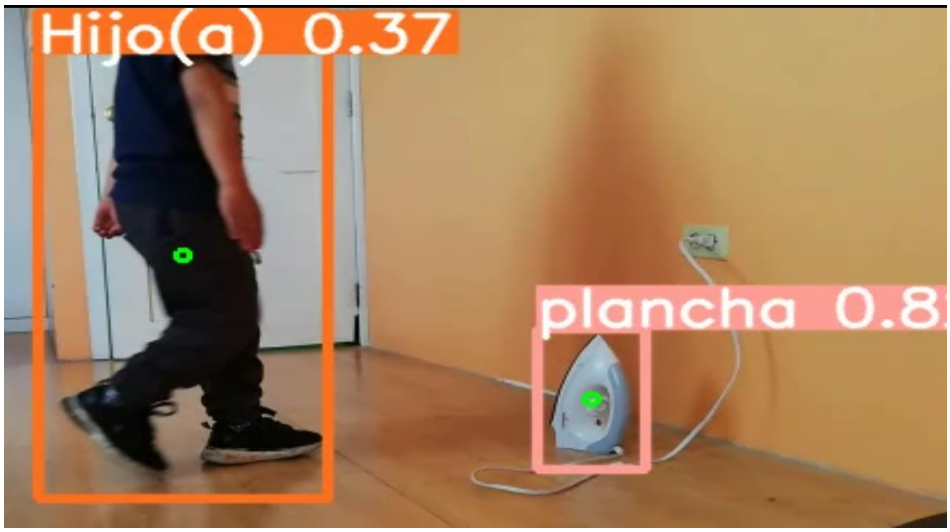


Ilustración 16-4: Distancia reducida, inicio de la emisión de alerta.

Realizado por: García, Jefferson, 2023.

Ilustración 17-4: Distancia reducida, inicio de la emisión de alerta.

Realizado por: García, Jefferson, 2023.

Es evidente que el porcentaje de certeza cambia con respecto a Yolo v5, pero no quiere decir que la alerta no se emita, el modelo en los dos escenarios ha interferido de manera tal que se evite un riesgo incluso sin la intervención de su tutor. Es prudente aclarar que el tutor siempre va a ser notificado por texto al igual que por la alerta sonora, el motivo de la alarma sonora no es asustar a los infantes, más bien su objetivo es llamar la atención del cuidador, dado el caso que no esté pendiente de las alertas escritas que se le envían a Telegram.

4.2.3. Efectividad de la emisión de alertas.

4.2.3.1. Alerta por texto.

La alerta por mensaje de texto llega de forma inmediata en cuanto la distancia entre el objeto y el infante es menor a 200 píxeles poniendo sobre aviso la proximidad del infante y el objeto de peligro.

La ilustración 18-4 evidencia los mensajes que el Bot de Telegram emite cuando el suceso de proximidad ocurre.

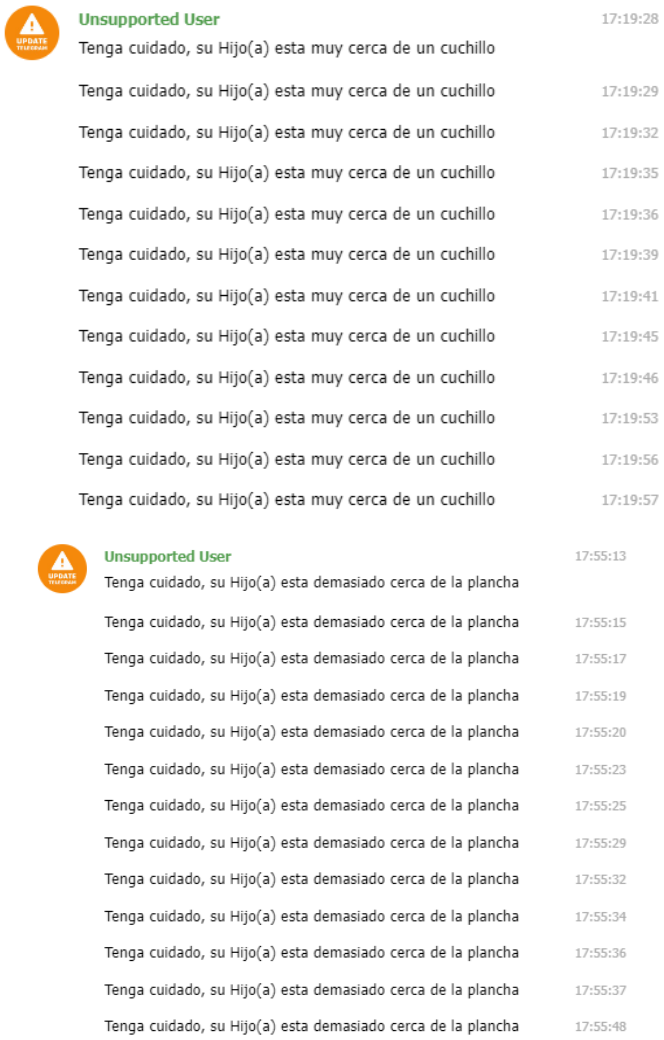


Ilustración 18-4: Alerta por mensaje de texto en la plataforma de Telegram.

Realizado por: García, Jefferson, 2023.

4.2.4. Criterio de padres, cuidadores y tutores bajo escala likert para la aplicabilidad y validación.

En forma de encuesta y bajo el criterio de diferentes responsables de infantes se valida el prototipo de manera que los resultados que se obtienen puedan estar acorde a lo que los usuarios necesitan

PREGUNTA 1

¿La aplicación de este prototipo resulta una herramienta de ayuda para el cuidado de niños de 1 a 5 años?

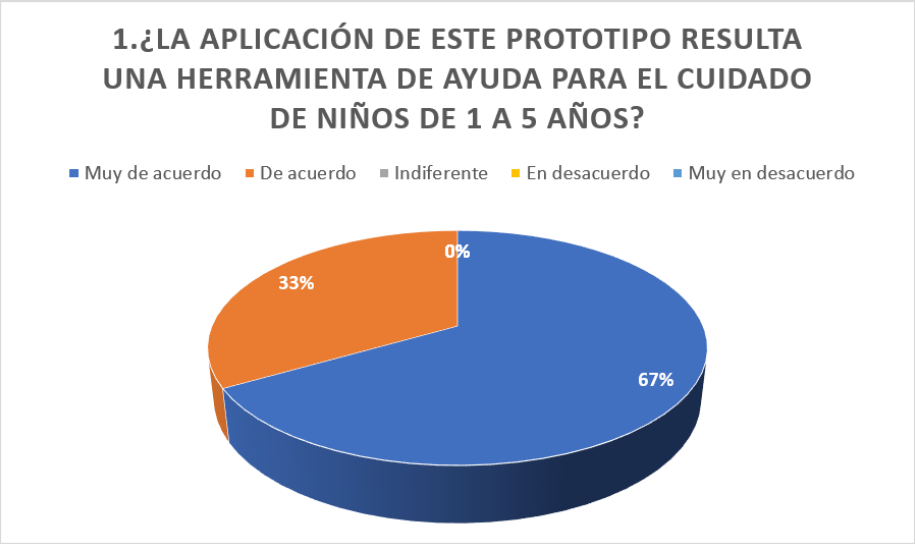


Ilustración 19-4: Resultados de la pregunta 1 en la Encuesta de Aplicabilidad

Fuente: Encuesta de Aplicabilidad

Realizado por: García, Jefferson, 2023.

Interpretación:

Como se observa en la ilustración 19-4 67% de encuestados están muy de acuerdo con la aplicación de este prototipo, ya que resulta una herramienta de ayuda para el cuidado de niños de 1 a 5 años, mientras que el 33% de encuestados están de acuerdo con su aplicación.

PREGUNTA 2

¿Cree usted que el diseño es bastante intuitivo y fácil de usar?

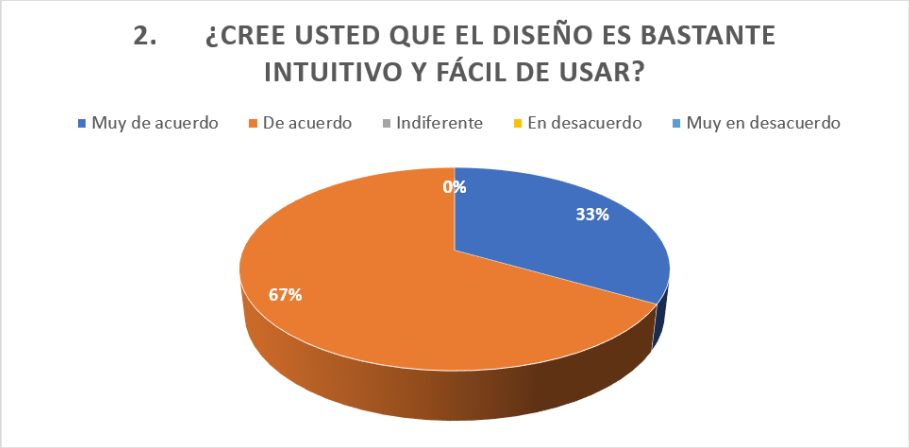


Ilustración 20-4: Resultados de la pregunta 2 en la Encuesta de Aplicabilidad

Fuente: Encuesta de Aplicabilidad

Realizado por: García, Jefferson, 2023.

Interpretación:

Como se puede observar en la ilustración 20-4, el 67% de los encuestados están de acuerdo que el diseño es bastante intuitivo y fácil de usar, mientras que el 33% de encuestados están muy de acuerdo con el diseño que se quiere implementar.

PREGUNTA 3

¿Cree usted que el número de objetos para poner en prueba el prototipo es suficiente?

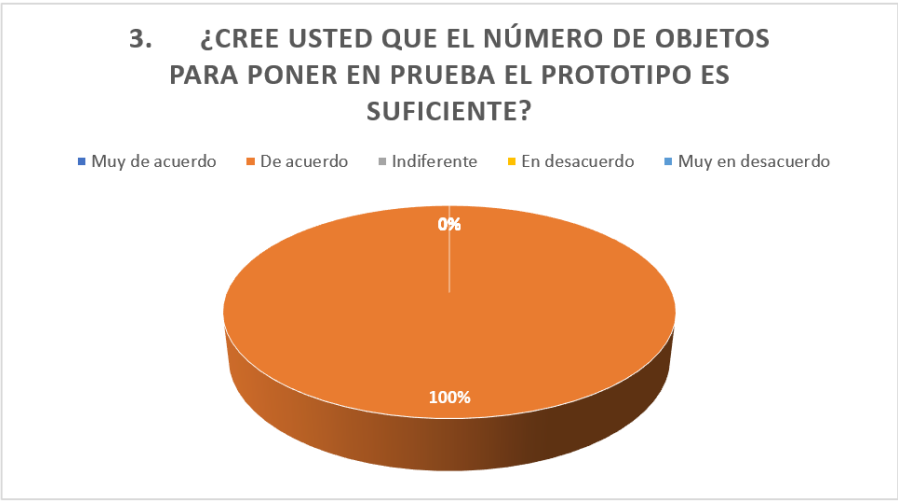


Ilustración 21-4: Resultados de la pregunta 3 en la Encuesta de Aplicabilidad

Fuente: Encuesta de Aplicabilidad

Realizado por: García, Jefferson, 2023.

Interpretación:

Como se puede observar en la ilustración 21-4, el 100% de encuestados están de acuerdo que es suficiente el número de objetos que se va a poner a prueba para observar la funcionalidad del prototipo.

PREGUNTA 4

¿Considera necesario realizar una capacitación para uso continuo de la aplicación?

Ilustración 22-4: Resultados de la pregunta 4 en la Encuesta de Aplicabilidad

Fuente: Encuesta de Aplicabilidad

Realizado por: García, Jefferson, 2023

Interpretación:

Como se puede observar en la ilustración 22-4, el 100 % de encuestados consideran que es necesario realizar una capacitación para uso continuo de la aplicación; debido a que se puede aprender mucho más acerca de los beneficios que ofrece el prototipo.

PREGUNTA 5

¿Cree usted que es prudente la alerta sonora?

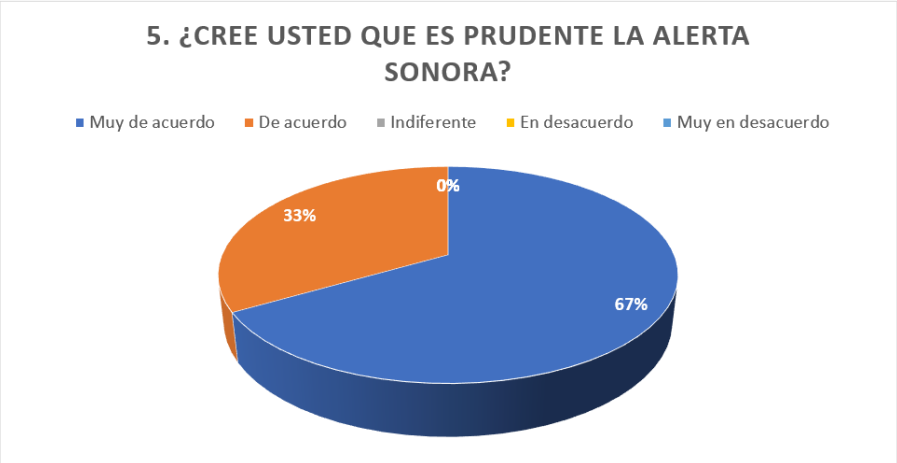


Ilustración 23-4: Resultados de la pregunta 5 en la Encuesta de Aplicabilidad

Fuente: Encuesta de Aplicabilidad

Realizado por: García, Jefferson, 2023.

Interpretación:

Como se puede observar en la ilustración 23-4, el 67% de los encuestados están muy de acuerdo con que es necesario la alerta sonora en caso de emergencia, mientras que un 33% de encuestados están también de acuerdo con que es necesario la alerta sonora; ya que puede evitar cualquier situación de peligro que esté expuesto el niño.

PREGUNTA 6

¿Considera que la experiencia al usar el aplicativo es satisfactoria?

Ilustración 24-4: Resultados de la pregunta 6 en la Encuesta de Aplicabilidad

Fuente: Encuesta de Aplicabilidad

Realizado por: García, Jefferson, 2023

Interpretación:

Como se puede observar en la ilustración 24-4, el 67% de los encuestados consideran que la experiencia de usar el aplicativo es satisfactoria , debido a que se puede brindar una respuesta oportuna, actuar rápidamente e identificar peligros que está expuesto el niño, mientras que el 33% de los encuestados están de acuerdo que la experiencia de usar el aplicativo es satisfactoria.

4.3. Presupuesto.

En la tabla 1-4 se detalla cada uno de los equipos, materiales y gastos utilizados en el desarrollo del presente trabajo, con un total de $2176.88 dólares americanos.

Tabla 1-4: Presupuesto del trabajo de titulación.

Artículo	Descripción	Cantidad	Valor unitario	Valor total
1.	Router repetidor	1	$17.88	$17.88
2.	teléfono Huawei P20 lite	1	$149.00	$149.00
3.	MSI Crosshair 15	1	$1455.00	$1455.00
4.	Xiaomi POCO X4 5G	1	$355.00	$355.00
5.	Varios	1	$200	$200
			Total	$2176.88

Realizado por: García, Jefferson, 2023.

CONCLUSIONES

- Se implementó un prototipo que cuida la integridad física de niños de 1 a 5 años frente la amenaza de cortaduras o quemaduras eminentes de objetos de peligro, mediante alertas sonoras y alertas vía mensaje.

- Se seleccionó la tecnología Yolo v5 para la detección en tiempo real por su facilidad de instalación, capacidad de detección superior a 30 cuadros por segundo, asignación de grado de confianza y delimitación de cuadros sobre los objetos.

- El porcentaje de falsos positivos en el modelo entrenado para la zona cocina es de 47% para la clase tenedor y para el caso de la zona habitación el valor es de 42% para la clase Hijo(a), siendo estos valores los más elevados para cuando el modelo reconoce un objeto que no se le presentó.

- El porcentaje de certeza del modelo pre entrenado de Yolo v5 es superior al 50% y para el modelo personalizado se obtiene una confianza superior al 25%. Es evidente que el porcentaje de certeza cambia con respecto al modelo original de Yolo v5, pero esto no implica que la alerta no se emita, el modelo en los dos escenarios ha interferido de manera tal que se evite un riesgo incluso sin la intervención de su tutor.

- Las pruebas realizadas con una ubicación de la cámara a 2 metros de altura y distancia hacen que la alerta se active cuando el niño guarda una relación menor a 300 píxeles que equivalen a 1m de distancia entre el objeto de peligro. En perspectiva si la distancia y altura de la cámara aumenta a un claro ejemplo de 3m y se conserva la relación de 300 píxeles las alertas se emiten cuando el niño guarda una relación de 1,32m de distancia al objeto de peligro.

- Iniciada la detección en tiempo real se obtiene un promedio de 54.78 FPS, y debido a que el envío de las alertas se las hace por internet se reduce a una tasa de 4 a 7 cuadros por segundo, por tanto, la transmisión de la alerta por la plataforma de Telegram ocurre con un retraso de 1 segundos, y de 2 segundos para la alerta sonora.

RECOMENDACIONES

- Al usar Yolo v5 por defecto las clases que viene incluidas están todas con nombres en inglés y no todas las clases corresponde a una zona en específico, por esto es necesario filtrar simplemente las clases de interés y cambiar la etiqueta correspondiente al idioma que se está empleando en el desarrollo.

- Visión artificial es una rama que representa un alto coste computacional, gráficos y de procesos centrales, por ello se recomienda correr el modelo sobre una tarjeta Nvidia.

- Se debe tener en cuenta que el tener una mayor distancia entre la cámara y los objetos pequeños reduce el grado de confianza con el que son detectados y la posibilidad de detectarlos.

- Se recomienda entrenar el modelo con más objetos que sean considerados de peligro dentro de una vivienda para el cuidado de la integridad del niño.

- Se le recomienda a la persona que va a estar a cargo del cuidado de los niños tener un plan de datos de internet y la aplicación de Telegram instalada en su dispositivo móvil para que las alertas lleguen de forma oportuna.

- Para mejorar el porcentaje de aprendizaje se debe incluir un número considerable de imágenes para tratar de cubrir en lo posible todos los eventos que se puedan realizar.

BIBLIOGRAFÍA

ANA, Gonzáles, FRANCISCO, Martínez y FERNANDO. 2006 *Técnicas y algoritmos básicos de visión artificial.* España : EDMANS. pp. 11-16.

ANACONDA. 2017**.** Conda. [Citado el: 27 de 12 de 2022.] Disponible en: https://docs.conda.io/en/latest/.

ANACONDA.DOCUMENTATION. 2023 *Instalación en Windows*. [Blog]. [Consulta: 12 enero 2023] Disponible en: https://docs.anaconda.com/anaconda/install/windows/.

ANALYTICS, Vidhya. 2023. *An Introduction to PyTorch – A Simple yet Powerful Deep Learning Library*. 2023. [Blog]. [Consulta: 10 enero 2023]. Disponible en: https://www.analyticsvidhya.com/blog/2018/02/pytorch-tutorial/.

ARIAS, José. 2020. *Proyecto de tesis, guia para la elaboración*. [En línea]. Arequipa-Perú. Depósito Legal en la Biblioteca Nacional del Perú N° 2020-05577. [Consulta: 11 enero 2023]. Disponible en: www.agogocursos.com.

BOCHKOVSKIY, Alexey, WANG, Chien-Yao y HONG, Yuan. 2020 YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv*, (Taiwan) pp. 3-10.

CALVO, Diego. 2022. *Redes neuronales artificiales.* [blog]. [Consulta: 29 diciembre 2022]. Disponible en: https://www.diegocalvo.es/clasificacion-de-redes-neuronales-artificiales/.

CARMILEMA, Edgar. 2017. Factores socioeconomicos y familiares asociados a riesgos de accidentes en el hogar en niños que llegan a la emergencia del hospital del dia del seguro social de Quito durante el segundo semestre del 2017. [En línea]. [Citado el: 21 de 12 de 2022.] Disponible en: http://repositorio.puce.edu.ec/bitstream/handle/22000/14984/TESIS%20FINAL%20ACCIDENTES%20EN%20EL%20HOGAR.pdf?sequence=1.

CHICAIZA, Fausto. 2017 *Sistema electrónico de alerta automatica para el reconocimienot de señales de tránsito reglamentarias, preventivas e informativas en la ciudad de Ambato.* 10 de Octubre de 2017. Disponible en: http://repositorio.uta.edu.ec/jspui/handle/123456789/26945.

CORDOVA, Wilson. 2020 *Implementacion de un sistema de reconocimiento de distanciamiento de distancia social como medida preventiva para COVID 19 usando deep learning.* Machala : s.n., 18 de diciembre de 2020. Disponible en: http://repositorio.utmachala.edu.ec/bitstream/48000/15982/1/TTFIC-2020-IS-DE00016.pdf.

DataCamp. 2022. *Explicación de la detección de objetos Yolo.* 2022. [Blog]. [Consulta: 27 diciembre 2022]. Disponible en: https://www.datacamp.com/blog/yolo-object-detection-explained.

DOMINGOS, Samara, ALVES, Nataly y LOPEZ, Maria. 2011. Hospitalizaciones por intoxicaciones de niños de cero a 14 años en un hospital del sur de Brasil. *SCIELO.* [En línea]. (Brasil). pp. 12.

ENGINEERS, LAST MINUTE. *Información sobre las características de ESP32 y su uso con Arduino IDE. 2023*. [Blog]. [Consulta: 14 enero 2023] Disponible en: https://lastminuteengineers.com/esp32-arduino-ide-tutorial/.

ESPEJEL, Omar. 2022. *Entendiendo PyTorch: las bases de las bases para hacer inteligencia artificial.* [Blog]. [Consulta: 11 septiembre 2022]. Disponible en: https://es.linkedin.com/pulse/entendiendo-pytorch-las-bases-de-para-hacer-omar-u-espejel.

FERNANDEZ, Samuel. 2022 *Huawei P20 Lite, análisis: el candidato a superventas de Huawei sube el listón fotográfico*. [Blog]. [Consulta: 11 enero 2023]. Disponible en: https://www.xatakamovil.com/analisis/huawei-p20-lite-analisis-review-con-caracteristicas-precio-y-especificaciones.

FIORENTINO, Jorge A, MOLISE, Claudia y STACH, Patricia. 2014. SCIELO. *Trauma en pediatría. Estudio epidemiológico en pacientes intenados en el hospital de niños Ricardo Gutierrez.* [En línea] pp 14-pp16.

GARCÍA, José. 2022. *Xiaomi POCO X4 Pro 5G, análisis: un claro aspirante a superventas con una batería de infarto*. [Blog]. [Consulta: 11 enero 2023]. Disponible en: https://www.xataka.com/analisis/xiaomi-poco-x4-pro-5g-analisis-caracteristicas-precio-especificaciones.

GERMÁN, Cuzco y ANTONIO, Layana. 2012. Sistema de alarma automatización y control de equipos a distancia a través de línea telefónica y págia web. (Trabajo de Titulación).

(Ingeniería). Universidad Politécnica Salesian, Facultad de Ingenierías. Guayaquil, Ecuador. pp. 15-16.

GONZALES, Jonathan. 2017. Reconocimiento de objetos utilizando OpenCV y Python en una raspberry pi 2 en una tlapalería. [En línea]. http://ri.uaemex.mx/bitstream/handle/20.500.11799/68150/tesis%20opencv%20python.pdf?sequence=1&isAllowed=y.

IBM. El modelo de redes neuronales. [En línea] 10 de Mayo de 2017. Disponible en: . https://www.ibm.com/docs/es/spss-modeler/SaaS?topic=networks-neural-model.

iPython. *Documentación de iPython.* [Blog]. [Consulta: 8 enero 2023]. Disponible en: https://ipython.readthedocs.io/en/stable/.

IRIUN. 2020. *Iriun webcam.* [Blog]. [Consulta: 10 enero 2023]. Disponible en: https://iriun.com/.

JUSTO, Alejandro. 2018**.** *Caracterización de los accidentes domésticos en niños menores de 5 años, comuna Dos Mangas. Centro de Salud San Antonio.* Guayaquil-Ecuador : (Trabajo de Titulación) (Maestría). Universidad Católica Santiago de Guayaquil, Facultad de Ciencias de la Salud. Guayaqui. 2018. pp. 11.

MAKESENSE. 2023. *MakeSense*. [En línea]. [Consulta: 12 septiembre 2022]. Disponible en: https://www.makesense.ai/.

MINCHOLA DE PEREZ, Adriana. 1984. *Accidentes en el hogar en niños menores de cinco años de Trujillo, Peru.* Trujillo : Bol of Sanit Panam, pp. 414-416.

MSI, Latam. 2023. *MSI CROSSHAIR 15.* 2023. [Blog]. [Consulta: 11 enero 2023]. Disponible en: https://latam.msi.com/Laptop/Crosshair-15-A11UX.

NUMPY. 2005. *Numpy.* 2005. [Consulta: 07 Enero 2023]. Disponible en: https://numpy.org/about/.

NUMPY. 2008. *¿Qué es NumPy?* 2008. [Consulta: 07 Enero 2023]. Disponible en: https://numpy.org/doc/stable/user/whatisnumpy.html.

OLIVA, Alejandro. 2018**.** Desarrollo de una aplicación de reconocimiento en. [En línea] 10 de Mayo de 2018. https://m.riunet.upv.es/bitstream/handle/10251/107243/OLIVA%20-%20Desarrollo%20de%20una%20aplicaci%C3%B3n%20de%20reconocimiento%20en%20im%C3%A1genes%20utilizando%20Deep%20Learning%20con%20O....pdf?sequence=1&isAllowed=y.

OpenCV. 2017. *OpenCV.* [blog]. [Consulta: 05 enero 2023]. Disponible en: https://opencv.org/about/.

OPENCV. 2018. *Información sobre la librería OpenCV.* 22 de Abril de 2018. Disponible en : https://opencv.org/.

MATPLOTLIB ORG. 2022. *matplotlib.* [blog]. [Consulta: 1 diciembre 2022]. Disponible en: https://matplotlib.org/.

PILLOW ORG. 2022. *PILLOW ORG.* [blog]. [Consulta: 10 diciembre 2023]. Disponible: https://pillow.readthedocs.io/en/stable/.

PYTORCH ORG. 2023**.** *PYTORCH ORG.* 2023. [Blog]. [Consulta: 10 enero 2023]. Disponible en: https://pytorch.org.

PYYAML ORG. 2023. *PyYaml org.* [blog]. [Consulta: 14 septiembre 2023]. Disponible en: https://pypi.org/project/PyYAML/.

REQUESTS ORG. 2023. *Requests org.* [blog]. [Consulta: 14 septiembre 2023]. Disponible en: https://pypi.org/project/requests/.

PANDAS, ORG. 2023. *Pandas.* [Consulta: 06 Enero 2023]. Disponible en: https://pandas.pydata.org/about/.

PONCE, Jefferson. 2017. *Estrategias educativas para prevenir los accidentes domiciliarios en niños de 4 años de la comunidad el Playon de San Francisco.* Tulcán-Ecuador : Disponible en: https://dspace.uniandes.edu.ec/bitstream/123456789/5547/1/TUTENF006-2017.pdf.

PRIETO, Guillermo. 2021. *Detección de víctimas mediante redes neuronales e imágenes térmicas.* Madrid : INDUSTRIALES, Disponible en: https://oa.upm.es/67546/1/TFG_GUILLERMO_PRIETO_SANCHEZ.pdf.

PyPi. 2023. *Psutil documentacion.* [Blog]. [Consulta: 10 enero 2023]. Disponible en: https://pypi.org/project/psutil/.

QUESTIONPRO. 2023. *Investigaicón Cualitativa.* [En línea]. [Consulta: 11 enero 2023]. Disponible en: https://www.questionpro.com/es/investigacion-cualitativa.html.

RAUL, Duque. 2011. *Python para todos.* España : pp. 7.

REDMON, Joseph y FARHADI, Ali. 2016. Yolo 9000. *arXiv*, (United State of America) pp. 2-4.

REDMON, Joseph y FARHADI, Ali. 2018. YOLOv3: An Incremental Improvement. *arXiv*, (United State of America) pp. 2-5.

REDMON, Joseph y SANTOSH, Divvala. 2016. You Only Look Once. *Cornell University.* (Washington) pp. 2-8.

SciPy ORG. 2023. *SCIPY.* [Blog]. [Consulta: 4 enero 2023].Disponible en: https://docs.scipy.org/doc/scipy/tutorial/general.html.

TECNIT. 2023. *ROUTER WIRELESS N300 D-LINK DIR-615 2.4GHz DOS ANTENAS.* [Blog]. [Consulta: 11 enero 2023] Disponible en: https://tecnit.com.ec/producto/router-wireless-n300-d-link-dir-615-2-4ghz-dos-antenas/.

TELEGRAM. 2023. *¿Qué es Telegram?* 2023. [Blog]. [Consulta: 10 enero 2023]. Disponile en: https://telegram.org.

Tqdm, org. 2010. *How to Use Progress Bars in Python?* [blog]. [Consulta: 14 septiembre 2010]. Dsiponible en: https://www.analyticsvidhya.com/blog/2021/05/how-to-use-progress-bars-in-python/.

VMWare, 2022. *VMWare Docs.* [blog]. [Consulta: 6 diciembre 2023]. Disponible en: https://docs.vmware.com/es/VMware-vSphere-Bitfusion/4.0/Example-Guide/GUID-74C3204C-744E-409A-B752-87177AD27E9E.html.

Waskom, Michael. 2012. seaborn. *Una introducción a seaborn.* [Consulta: 07 Enero 2023]. Disponible en: https://seaborn.pydata.org/tutorial/introduction.

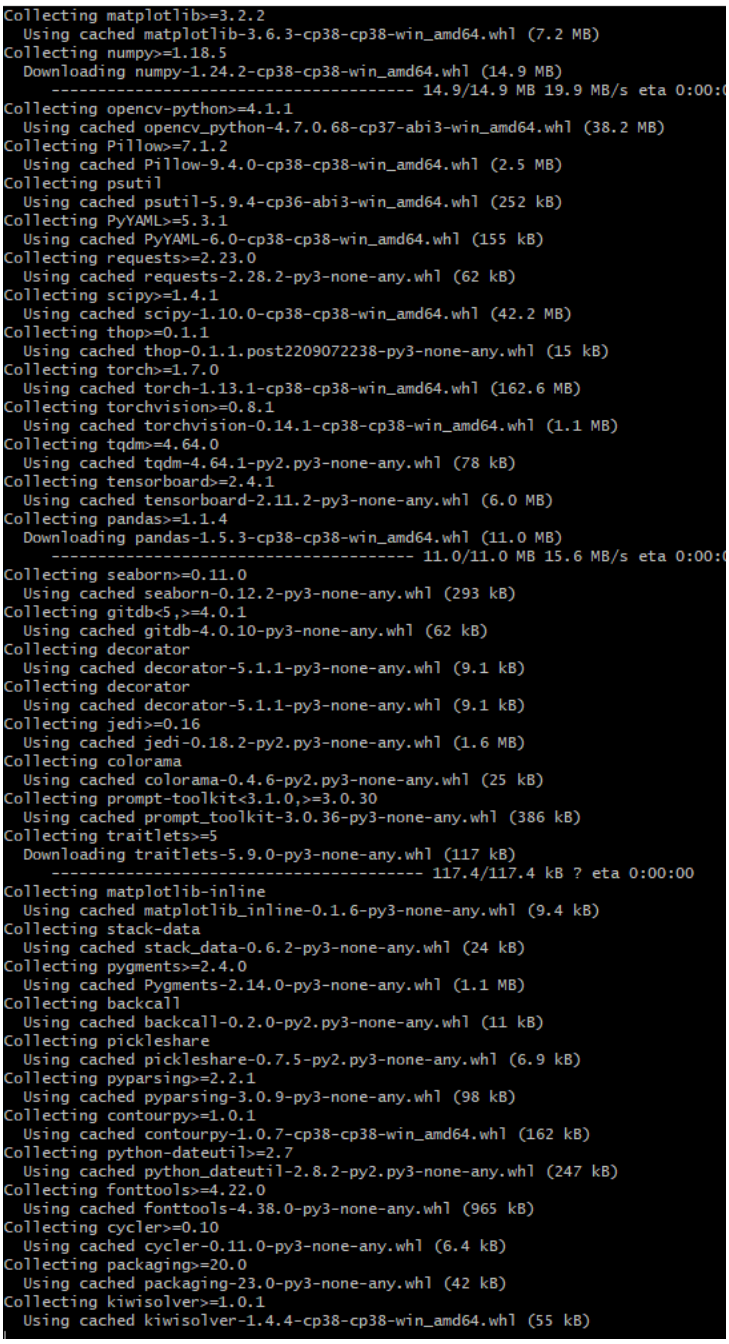
YAO, Jin, QI, Jiaming y ZHANG, Jie. 2021. A Real-Time Detection Algorithm for Kiwifruit Defects Based on YOLOv5. *Electronics*, (China) pp. 3-4.

YI LING, Tsung, MAIRE, Michael y BELONGIE, Serge. Microsoft COCO: Common Objects in Context. *arXiv.* (2015). (Estados Unidos). pp. 2-8.

ANEXOS

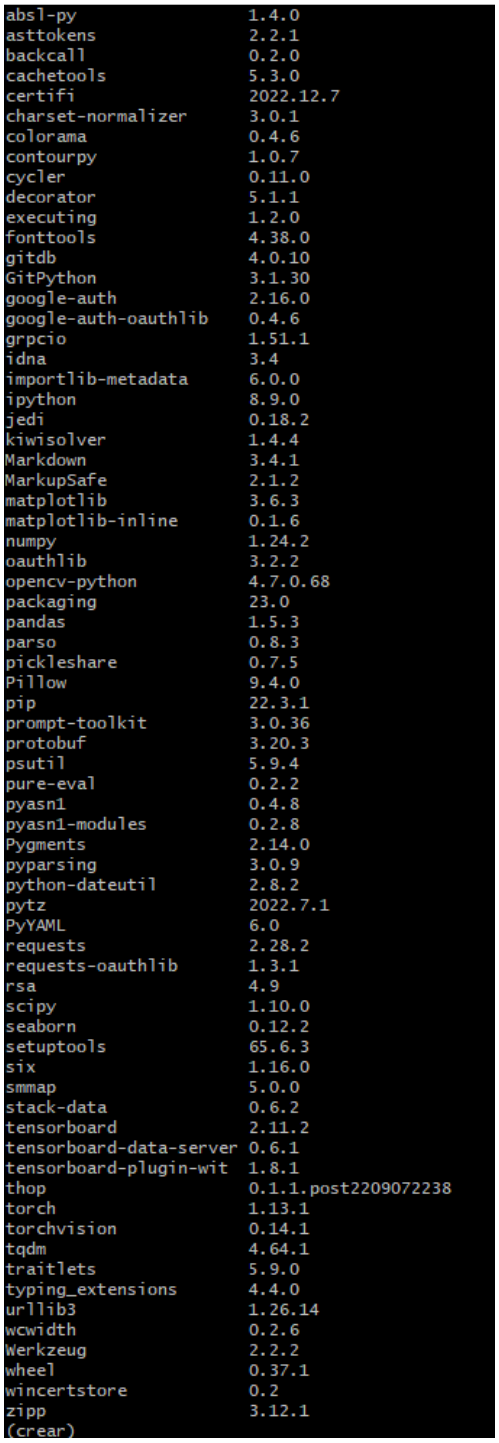
Anexo A: Instalación de las librerías.

Para simplificar todo el proceso se implementó un archivo denominado requeriments.tx que contiene cada librería que se necesita para el buen funcionamiento del prototipo dentro del ambiente creado, lo único que se debe de hacer es correr el siguiente comando *pip install -r requeriments.tx* en su efecto se mostrará el siguiente proceso.



Anexo B: Verificación de las librerías necesarias.

Para constatar solo se tiene que correr el comando *pip list.*

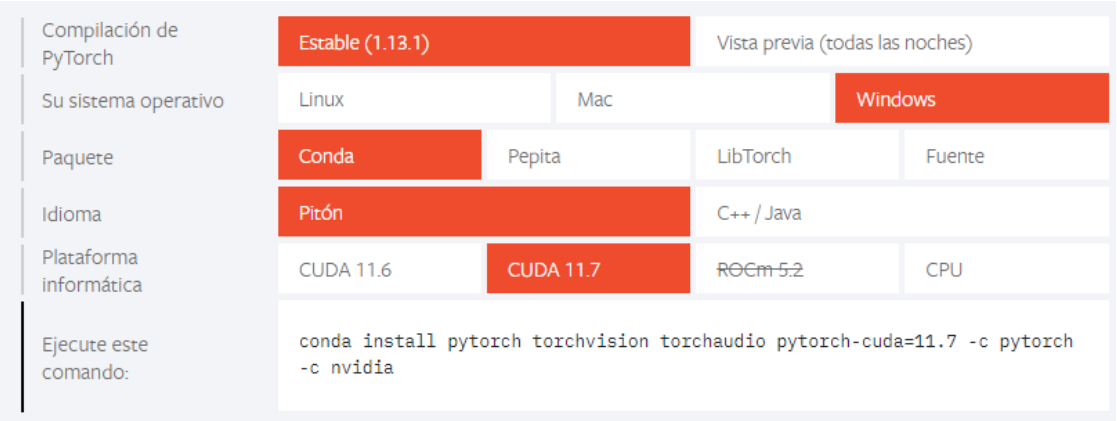


```
absl-py                 1.4.0
asttokens               2.2.1
backcall                0.2.0
cachetools              5.3.0
certifi                 2022.12.7
charset-normalizer      3.0.1
colorama                0.4.6
contourpy               1.0.7
cycler                  0.11.0
decorator               5.1.1
executing               1.2.0
fonttools               4.38.0
gitdb                   4.0.10
GitPython               3.1.30
google-auth             2.16.0
google-auth-oauthlib    0.4.6
grpcio                  1.51.1
idna                    3.4
importlib-metadata      6.0.0
ipython                 8.9.0
jedi                    0.18.2
kiwisolver              1.4.4
Markdown                3.4.1
MarkupSafe              2.1.2
matplotlib              3.6.3
matplotlib-inline       0.1.6
numpy                   1.24.2
oauthlib                3.2.2
opencv-python           4.7.0.68
packaging               23.0
pandas                  1.5.3
parso                   0.8.3
pickleshare             0.7.5
Pillow                  9.4.0
pip                     22.3.1
prompt-toolkit          3.0.36
protobuf                3.20.3
psutil                  5.9.4
pure-eval               0.2.2
pyasn1                  0.4.8
pyasn1-modules          0.2.8
Pygments                2.14.0
pyparsing               3.0.9
python-dateutil         2.8.2
pytz                    2022.7.1
PyYAML                  6.0
requests                2.28.2
requests-oauthlib       1.3.1
rsa                     4.9
scipy                   1.10.0
seaborn                 0.12.2
setuptools              65.6.3
six                     1.16.0
smmap                   5.0.0
stack-data              0.6.2
tensorboard             2.11.2
tensorboard-data-server 0.6.1
tensorboard-plugin-wit  1.8.1
thop                    0.1.1.post2209072238
torch                   1.13.1
torchvision             0.14.1
tqdm                    4.64.1
traitlets               5.9.0
typing_extensions       4.4.0
urllib3                 1.26.14
wcwidth                 0.2.6
Werkzeug                2.2.2
wheel                   0.37.1
wincertstore            0.2
zipp                    3.12.1
(crear)
```

Y será todas las librerías con las que el programa puede ser funcional.

Anexo C: Instalación de Pytorch para ejecutar los procesos en GPU.

En la página oficial de Pytorch solo se tiene que optar por las opciones que en ese momento sus recursos le permitan. En este caso dadas las especificaciones de nuestro procesador se elige la siguiente opción.



Por tanto, el comando que se tiene que ejecutar es *conda install pytorch torchvision torchaudio pytorch-cuda=11.7 -c pytorch -c Nvidia*

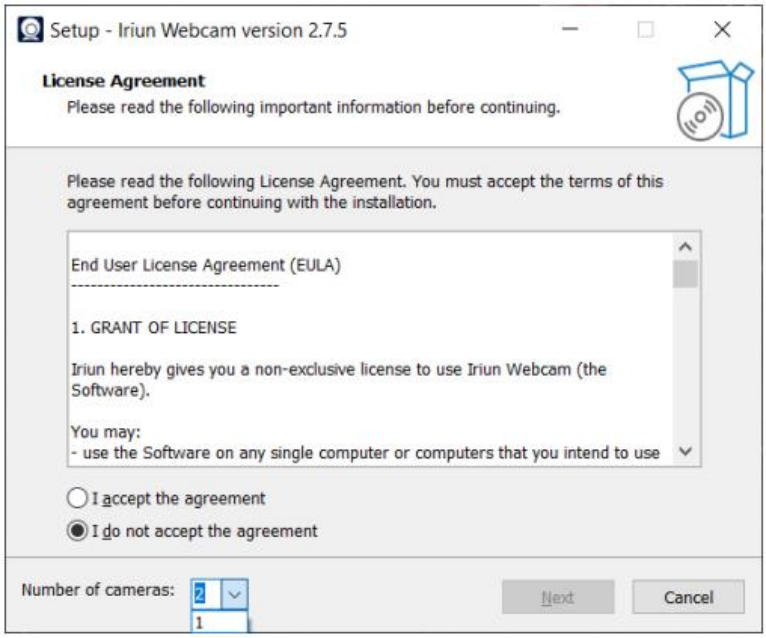
Anexo D: Instalación de Iriun.

Para el uso de las cámaras se puede acceder a las cámaras de los dispositivos móviles de manera inalámbrica, permitiéndonos tener facilidad de uso y mejores resoluciones.

Para eso se sigue el proceso presentado.

Descargar iriun en el computador y en los dispositivos móviles, desde el siguiente enlace: https://iriun.com

En la opción “Number of cameras” seleccionamos el número de cámaras que necesita instalar.



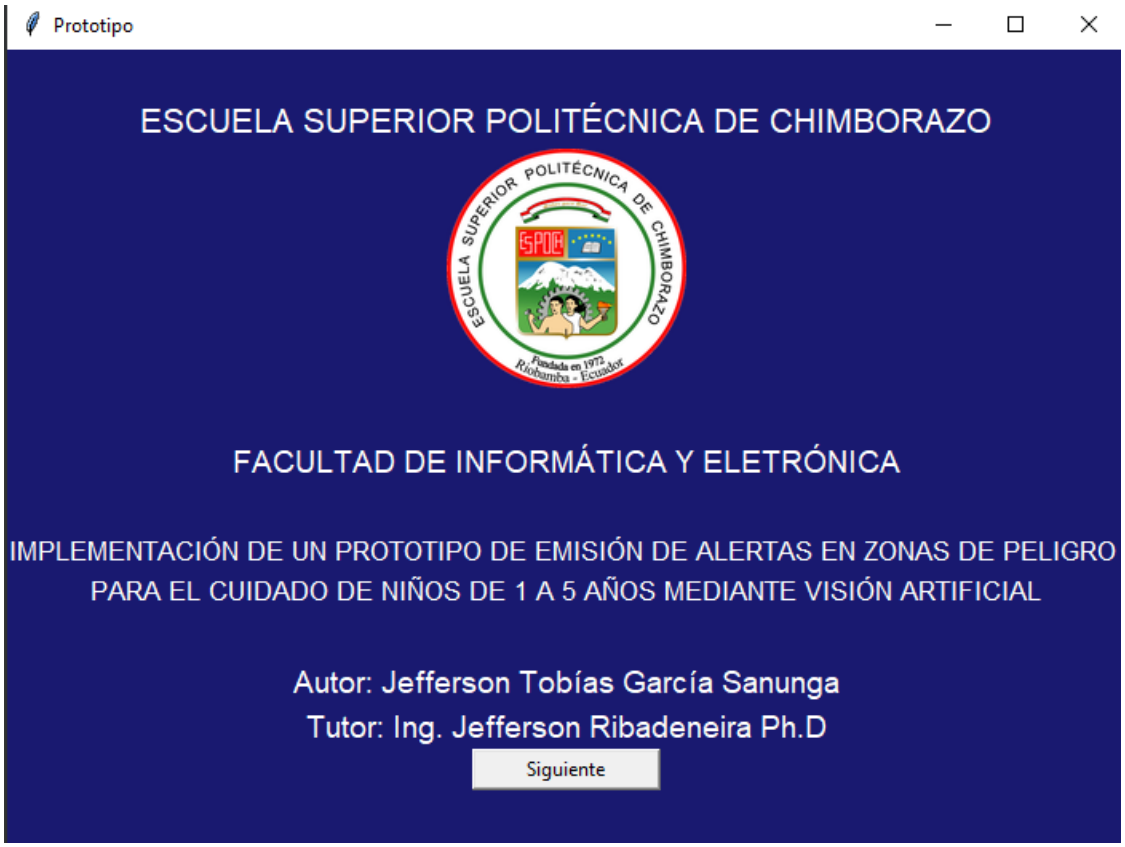
Para realizar la conexión se debe abrir la aplicación "iriun" en el computador y en los dispositivos móviles.

Para conectar las cámaras, se debe hacer clic en el botón "Continuar" y se conectara automáticamente las cámaras de los dispositivos móviles como webcams.

Anexo E: Funcionalidad del prototipo.

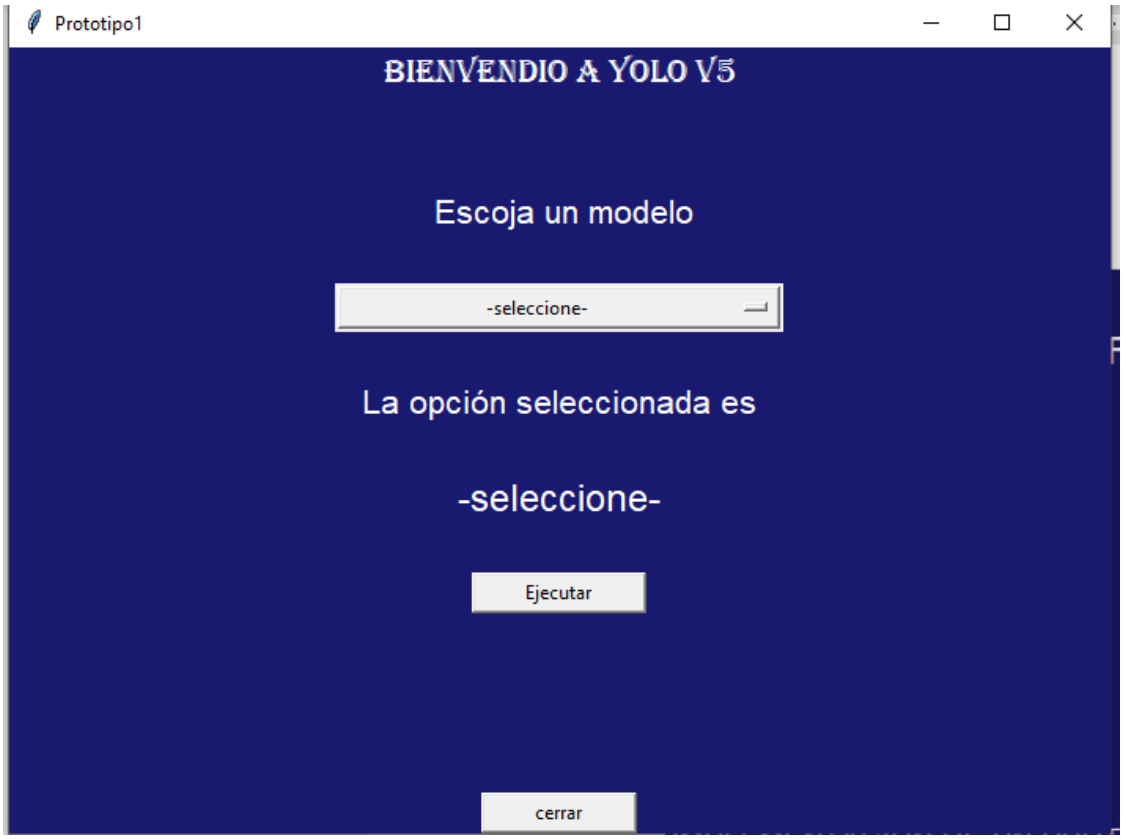
Se logra empaquetar cada uno de los archivos de forma tal que se pueden ejecutar con un solo archivo ejecutable bastante intuitivo.

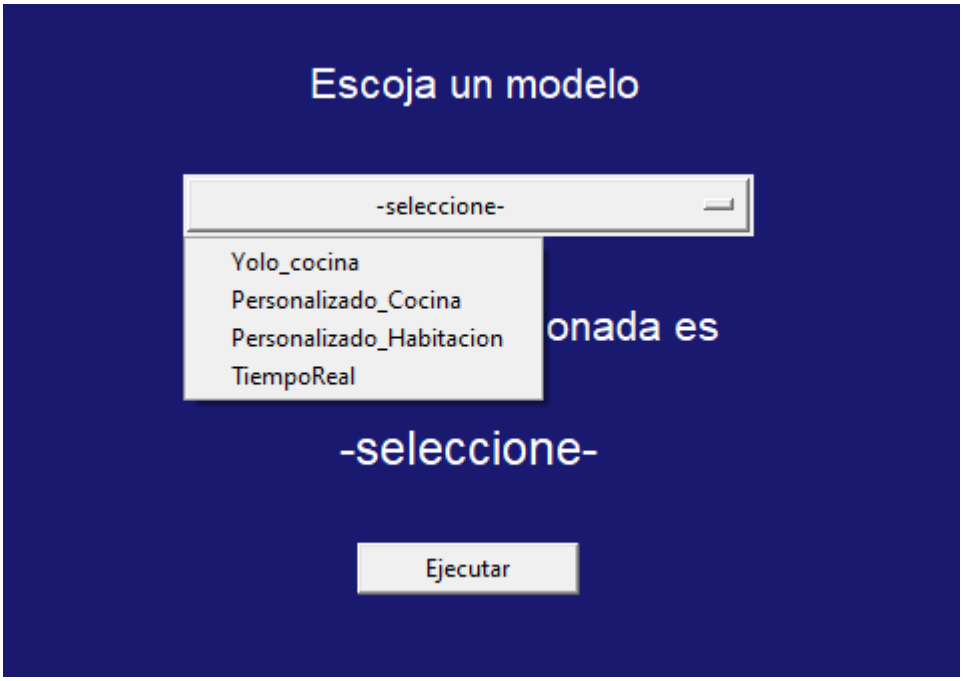
La primera ventana será en efecto la bienvenida al prototipo como se presenta a continuación



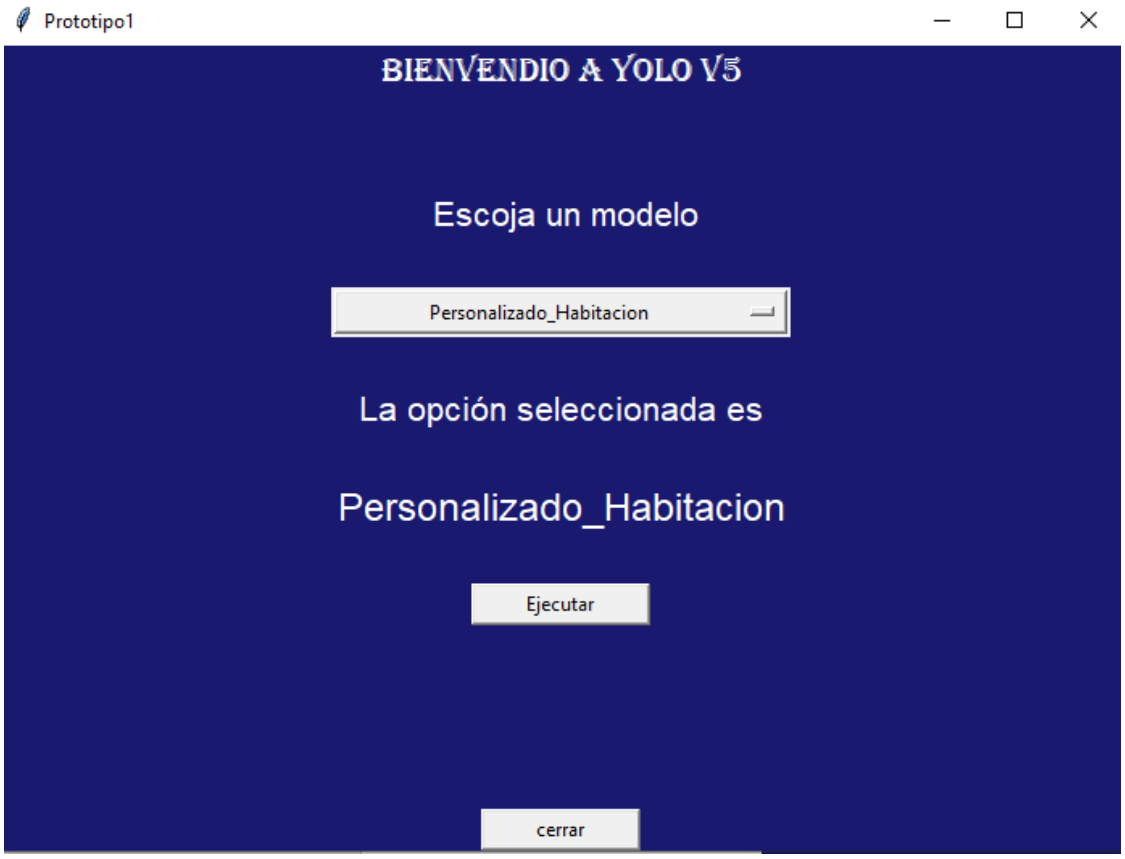
Solo hay que darle en siguiente y se solicitará al usuario que elija la zona en donde quiere detectar.

Por su parte el programa tiene varios modelos entrenados y su nombre indica que lugar es el que quieren efectuar el proceso de detección

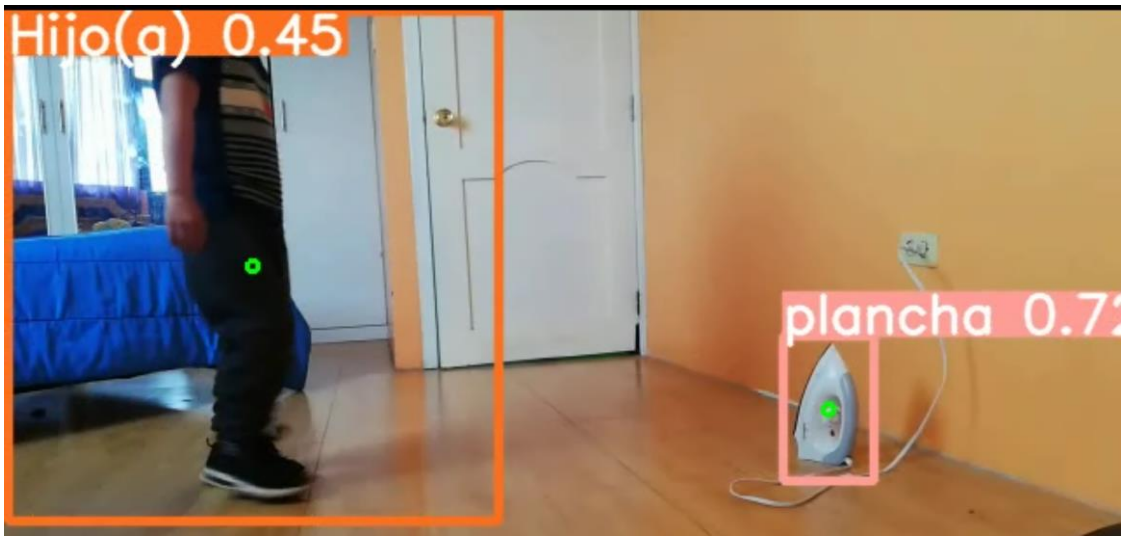


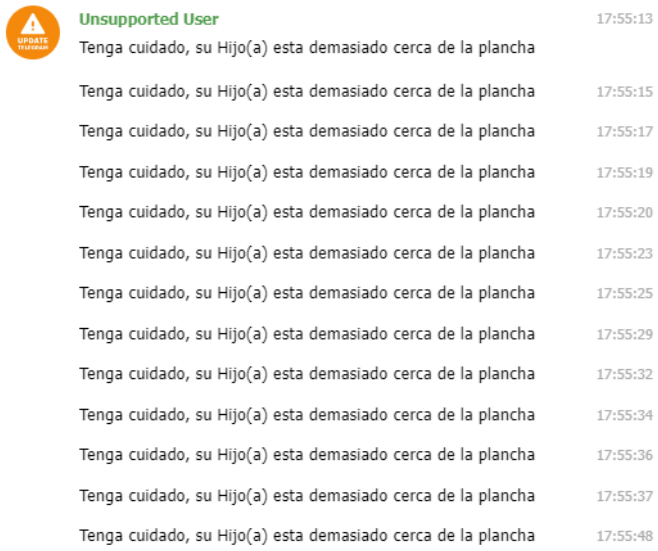


Si se selección habitación en este caso, la conectividad será a la cámara que se encuentre en la habitación permitiendo censar en todo momento la actividad de su hijo respecto a los objetos de peligro

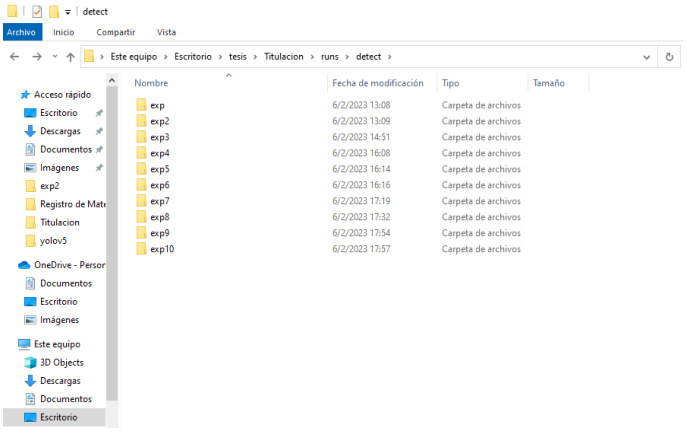


Se elije ejecutar y el proceso será automático.





Cada proceso, cada detección se guarda en la carpeta runs/detect/exp para tener una idea de todo el proceso o los procesos ejecutados.



Anexo F: Encuesta de aplicabilidad

Pregunta 1.

¿La aplicación de este prototipo resulta una herramienta de ayuda para el cuidado de niños de 1 a 5 años?

a) Muy de acuerdo

b) De acuerdo

c) Indiferente

d) En desacuerdo

e) Muy en desacuerdo

Pregunta 2.

¿ Cree usted que el diseño es bastante intuitivo y fácil de usar?

a) Muy de acuerdo

b) De acuerdo

c) Indiferente

d) En desacuerdo

e) Muy en desacuerdo

Pregunta 3.

¿ Cree usted que el número de objetos para poner en prueba el prototipo es suficiente?

a) Muy de acuerdo

b) De acuerdo

c) Indiferente

d) En desacuerdo

e) Muy en desacuerdo

Pregunta 4.

¿ Considera necesario realizar una capacitación para uso continuo de la aplicación?

a) Muy de acuerdo

b) De acuerdo

c) Indiferente

d) En desacuerdo

e) Muy en desacuerdo

Pregunta 5.

¿ Cree usted que es prudente la alerta sonora?

a) Muy de acuerdo

b) De acuerdo

c) Indiferente

d) En desacuerdo

e) Muy en desacuerdo

Pregunta 6.

¿ Considera que la experiencia al usar el aplicativo es satisfactoria?

a) Muy de acuerdo

b) De acuerdo

c) Indiferente

d) En desacuerdo

e) Muy en desacuerdo

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

DIRECCIÓN DE BIBLIOTECAS Y RECURSOS DEL APRENDIZAJE

UNIDAD DE PROCESOS TÉCNICOS

REVISIÓN DE NORMAS TÉCNICAS, RESUMEN Y BIBLIOGRAFÍA

Fecha de entrega: 17/ 04 / 2023

INFORMACIÓN DEL AUTOR
Jefferson Tobías García Sanunga
INFORMACIÓN INSTITUCIONAL
Facultad: Informática y electrónica
Carrera: Telecomunicaciones
Título a optar: Ingeniero en Telecomunicaciones
f. Analista de Biblioteca responsable: Ing. Fernanda Arévalo M.

UNIDAD DE PROCESOS TÉCNICOS
DBRA
DIRECCIÓN DE BIBLIOTECAS Y RECURSOS DEL APRENDIZAJE
ESPOCH