"2023. Año del Septuagésimo Aniversario del Reconocimiento del Derecho al Voto de las Mujeres en México".

Uso de inteligencia artificial como herramienta para predicción de accidentes en los hogares para niños menores de 12 años

TESIS

QUE PARA OBTENER EL GRADO DE:

MAESTRO EN TECNOLOGÍAS DE LA INFORMACIÓN

PRESENTA:

Ing. Marco Antonio García Téllez

DIRECTOR DE TESIS

Dr. Omar Barragán Pérez

CUAUTITLÁN IZCALLI, EDO. DE MÉXICO. **Enero, 2023**

"2023. Año del Septuagésimo Aniversario del Reconocimiento del Derecho al Voto de las Mujeres en México".

Cuautitlán Izcalli, Estado de México a 09 de febrero de 2023

TESCI/DIDT/10/II/23

DIRECCIÓN ACADÉMICA
DEPARTAMENTO DE INVESTIGACIÓN Y DESARROLLO TECNOLÓGICO
COORDINACIÓN DE POSGRADO

INGENIERO
MARCO ANTONIO GARCÍA TÉLLEZ
P R E S E N T E

Por este conducto me permito informarle que puede proceder a la digitalización del Trabajo de Tesis titulado:

"USO DE INTELIGENCIA ARTIFICIAL COMO HERRAMIENTA PARA PREDICCIÓN DE ACCIDENTES EN LOS HOGARES PARA NIÑOS MENORES DE 12 AÑOS"

Ya que la comisión encargada de revisar el trabajo que se presenta para efectos de titulación, han dado su autorización conforme a lo estipulado en el Lineamiento para la operación de los Estudios de Posgrado en el Sistema Nacional de Institutos Tecnológicos.

Sin nada más que agregar, quedo a sus órdenes para cualquier aclaración.

A T E N T A M E N T E

MTRA. ROCÍO ORTEGA JIMÉNEZ
DEPARTAMENTO DE
INVESTIGACIÓN Y DESARROLLO TECNOLÓGICO
COORDINACIÓN DE POSGRADO

c.c.p. Archivo
Departamento de Titulación
Expediente del alumno

Contenido

Capítulo I. Introducción

Los accidentes en el hogar son algo común, estos pueden ocasionar daños leves y en algunos casos pueden llegar a ser muy graves, la mejor solución a este problema es la prevención, es indispensable tener especial cuidado con los niños y adultos mayores. Tener un adecuado orden de los objetos como, por ejemplo: utensilios de cocina punzantes, herramientas, juguetes, medicamentos, productos químicos, bebidas alcohólicas, es primordial para minimizar este tipo de accidentes y, sobre todo, tener conciencia que cualquier acción que se realice, puede ocasionar alguna eventualidad no esperada, por ello, siempre se tiene que ser precavido al momento de realizar cualquier tarea doméstica o actividad dentro del hogar.

En el hogar es donde se es más propenso a sufrir algún tipo de accidente, debido a que es donde se pasa la mayor parte del tiempo y actualmente, debido a los efectos que ha provocado la pandemia del 2020, muchas personas han tenido la oportunidad de trabajar desde casa, de igual manera los niños y adolescentes toman sus clases desde casa, por lo que el tiempo que se pasa en casa se ha incrementado considerablemente, dicho lo anterior, es recomendable que el lugar dedicado al trabajo y al estudio en casa este limpio y ordenado, para generar un ambiente más confortable y seguro.

El propósito de esta investigación es prevenir algunos de estos accidentes que sufren los niños menores de 12 años dentro del hogar, mediante la visión por computadora y la inteligencia artificial, desarrollar y optimizar algoritmos de machine learning que sean lo suficientemente capaces de detectar oportunamente un incidente.

Dicho propósito se puede obtener mediante el entrenamiento de datos etiquetados, es decir, imágenes etiquetadas donde aparezcan objetos que puedan representar algún riesgo y a niños acercándose a estos objetos potencialmente riesgosos como es el caso de: cocinas, objetos punzo cortantes, herramientas, contactos de luz, etc. mediante una cámara de video se puede realizar el monitoreo de los niños, analizar el video y tratar de identificar alguna situación de riesgo donde los niños se puedan ver involucrados y en cuanto se detecte una situación riesgosa, generar alertas mediante él envió de notificaciones al smartphone del adulto o tutor.

Objetivos

Los accidentes en niños dentro del hogar es una problemática a gran escala que existe actualmente. Derivado de la pandemia del 2020 muchos

niños se quedaron en resguardo dentro de su hogar, por lo que el número de casos se incrementó durante este periodo. Debido a este incremente en los accidentes, es importante generar sistemas de inteligencia artificial que ayude reducir las estadísticas y prevenir los accidentes de los niños dentro del hogar para salvaguardar la integridad de los niños. Este tipo de eventualidades se pueden detectar oportunamente mediante la visión por computadora y la implementación de algoritmos de inteligencia artificial, los cuales previamente entrenados pueden detectar este tipo de eventualidades para ayudar a disminuir este tipo de accidentes.

Un hogar que está siendo monitoreado constantemente mediante inteligencia artificial, es capaz de tomar decisiones cuando se detecten patrones o situaciones que puedan representar un riesgo para un niño. El tipo de decisiones que puede tomar la inteligencia artificial van desde enviar un simple mensaje de notificación al teléfono móvil de los padres o tutor, hasta activar mecanismos de seguridad como, por ejemplo: activar cerraduras, cortar el suministro de energía eléctrica, activar robots de limpieza, etc. Para que de esta forma se puede prevenir alguna situación lamentable.

Objetivos específicos

- Desarrollar un sistema que detecte oportunamente alguna eventualidad de riesgo o accidente dentro de los hogares de los niños, mediante la visión por computadora y la inteligencia artificial para disminuir las estadísticas sobre las lesiones que sufren los niños en el hogar.
- Investigar cuales son los lugares, objetos y situaciones más comunes que ocasionan accidentes dentro de los hogares de los niños.
- Diseñar y entrenar algoritmos de machine learning para reconocer objetos y personas en las imágenes y videos.
- Instalar e implementar el equipo de video para detectar dentro del hogar objetos que representen algún riesgo para las personas.
- Validar y analizar los resultados obtenidos para mejorar y optimizar los algoritmos de machine learning.

Justificación

Existe un riesgo latente de que cualquier niño sufra un accidente dentro de su propio hogar, puede existir una serie de circunstancias, eventos y objetos que ocasionan este tipo de incidentes, entre ellos se puede mencionar: escaleras, estufas, contactos de luz eléctrica, mascotas, albercas, utensilios de cocina punzo cortantes, herramientas, medicamentos, flamas, productos

químicos, bebidas alcohólicas u objetos tirados en el suelo, etc. Dichas eventualidades dañan la integridad física del infante además de pérdidas materiales y monetarias, por lo que el prevenir estas situaciones es de vital importancia para el buen desarrollo de los niños.

Muchos de los accidentes que se originan en el hogar se pueden prevenir siempre y cuando los padres o tutores estén al pendiente de los infantes, sin embargo, esto no siempre es posible y basta un pequeño descuido, para originar una desgracia, en la mayoría de los casos se deja a los niños al cuidado de adultos mayores, porque los padres tienen la necesidad de salir a trabajar para generar ingresos y realmente los padres no tienen el conocimiento de donde se encuentran o que están haciendo sus hijos, por esta razón es indispensable crear un sistema que este monitoreando constantemente la actividad de los niños para prevenir alguna eventualidad no esperada.

Los algoritmos de inteligencia artificial y la visión por computadora han mejorado de tal manera que implementar estos algoritmos o frameworks resulta fácil y no es necesario contar con un equipo de cómputo sofisticado o especializado para este propósito. El sistema que se va a desarrollar está basado en redes neuronales que mediante el entrenamiento de modelos donde el dataset será el tipo de imágenes que se pretende detectar, por ejemplo: cuando los niños se acercan a la cocina, herramientas, contactos de luz, medicamentos que se encuentran al alcance de los niños o cualquier otra eventualidad antes mencionada que pueda representar algún riesgo.

Actualmente existe una gran cantidad de sistemas cuyo propósito es el reconocimiento facial y la detección de objetos comunes en videos, pero están más enfocados al ámbito público, es decir, en la conducción autónoma, como el automóvil Tesla que mediante cámaras de video el sistema puede reconocer objetos como por ejemplo: otros automóviles, peatones y obstáculos que pueden afectar la conducción del vehículo autónomo, en las plazas comerciales y en lugares concurridos como los aeropuertos ya es común observar sistemas que detectan el número de personas que se encuentran a una hora determinada y hasta la implementación de reconocimiento facial para identificar personas que posiblemente sean buscadas por alguna organización de seguridad internacional.

La finalidad de implementar este sistema de inteligencia artificial dentro de los hogares es tener un constante monitoreo de los niños para asegurar su integridad y reducir las estadísticas de los accidentes que ocurren dentro de los hogares de los niños.

Planteamiento de la hipótesis

Recientemente el auge de la inteligencia artificial ha crecido exponencialmente debido a que ahora es más barato conseguir hardware más poderoso en comparación hace apenas un par de años y gracias a esto, se ha logrado probar y optimizar los algoritmos de machine learning para mejorar la precisión de sus resultados. Esto ha facilitado la implementación de inteligencia artificial y la creación de prototipos funcionales para diferentes áreas de investigación. Una de estas áreas de investigación en la cual se pretende enfocar este proyecto es en la seguridad dentro de los hogares, sobre todo en el monitoreo de las actividades de los niños o infantes, para salvaguardar su integridad física ante un posible accidente.

Lo interesante de este tipo de investigación es desarrollar algoritmos de machine learning que sean lo suficientemente capaces de detectar este tipo de anomalías dentro del hogar. Uno de los principales pasos es, entrenar un modelo de machine learning con imágenes donde aparezcan objetos que puedan representar algún riesgo, personas adultas o niños acercándose a objetos potencialmente riesgosos y mediante una cámara de video con la que se va a realizar el monitoreo, capturar el video y analizar cada frame dentro del video para reconocer dentro del frame alguna situación de riesgo en que las personas o los niños puedan verse involucrados, cuando se detecte alguna situación eventualmente riesgosa generar alguna alerta o activar algún mecanismo de seguridad para evitar un posible accidente.

Hipótesis

El presente trabajo de investigación pretende demostrar que, mediante la inteligencia, la visión por computadora y con una poca inversión en equipos de cómputo y video se puede desarrollar un sistema que sea capaz de detectar y prevenir oportunamente accidentes de niños dentro de los hogares. Durante la implementación del proyecto se propone utilizar algoritmos de machine learning que no consuman demasiados recursos de computacionales, esto con la finalidad de poder hacer una implementación en equipos de cómputo que no sean especializados y optimizados en cuanto al manejo de GPU (Graphics Processing Unit), debido a que este es un recurso que se suele utilizar con gran intensidad al momento de procesar imágenes y videos, por ello es necesario elegir los algoritmos optimizados y livianos para no alterar o deteriorar la precisión del reconocimiento de objetos. Otro punto fundamental es la generación de alertas y la activación de algún mecanismo de seguridad, una vez que se detecta la situación o el evento de riesgo, en necesario notificar en primera instancia a los padres de familia o tutores que estén a cargo en ese momento del infante, para que ellos tomen medidas precautorias y prevenir

alguna situación posterior y como segundo nivel o complemento es activar alguna medida de seguridad como por ejemplo bloquear el suministro de luz eléctrica, activar un robot de limpieza, etc.

Capítulo II. Marco Teórico y Estado del Arte

Marco Teórico

Machine learning

Tradicionalmente un programador de computadora da instrucciones paso a paso a una computadora para resolver una tarea en una aplicación de un área en particular, sin embargo, hay situaciones donde el tradicional enfoque de programación se dificulta y el enfoque de machine learning se vuelve útil en instruir a la computadora.

A través del aprendizaje, un programa de computadora obtiene nueva información y altera su proceso de toma de decisiones existente, realizando así mejor una tarea. La esencia de Machine Learning es que el algoritmo de aprendizaje crea nuevas reglas o pasos. de los datos y resuelve un problema. Mediante el uso de diferentes conjuntos de datos, el mismo aprendizaje algoritmo se puede utilizar para realizar diferentes tareas. Por ejemplo, puede utilizar el mismo algoritmo de aprendizaje para detectar un correo electrónico no deseado, así como para predecir el cambio de precio en la bolsa de Valores.

Machine learning es un campo de la informática que hace uso de algoritmos y estadísticas métodos para aprender de los datos, hacer inferencias y reconocer patrones sin programación explícita. Es un enfoque para desarrollar un sistema de software que aprende de los datos y se adapta para mejorar su capacidad de realizar tareas, como la predicción (Dash & Kumar, 2022).

Deep learning

Es un subconjunto de Machine Learning que utiliza un sistema neuronal de varias capas red para imitar el complejo procesamiento de información del cerebro humano. Similar a las neuronas biológicas en el cerebro humano, un perceptrón se utiliza como el edificio básico bloque de una red neuronal artificial y realiza el procesamiento de la información. Una red neuronal consta de un gran número de neuronas (o perceptrones) interconectadas con entre sí en capas y es responsable del procesamiento paralelo masivo de datos de entrada datos para generar salidas. Una red neuronal consta de varias capas: una capa de entrada, una capa de salida y varias capas ocultas.

En un sistema de aprendizaje profundo, lo oculto, las capas deciden la profundidad de la red. El número de capas ocultas depende de la naturaleza del problema en cuestión y el volumen de datos considerados para el

procesamiento. Una red de aprendizaje profundo utiliza enfoques de aprendizaje tanto supervisados como no supervisados para el entrenamiento de la red neuronal. Una variedad de arquitecturas y aprendizaje supervisado Los algoritmos se utilizan en el aprendizaje profundo, siendo el más común Red Neuronal Recurrente (RNN) y Red Neural Convolucional (CNN) (Dash & Kumar, 2022)

Redes neuronales

Una Red de Neuronas Artificiales (RNA) es un paradigma de procesamiento de información inicialmente inspirado en el modo en el que lo hace el cerebro. El elemento clave de este paradigma es su estructura. Las RNA están compuestas por un cierto número de elementos de procesamiento o neuronas que trabajan al unísono para resolver un problema específico. Las redes neuronales actuales se basan en el modelo matemático de neurona propuesto por McCulloch y Pitts en 1943 (McCulloch y Pitts, 1943).

En dicho modelo (Figura 1) cada neurona recibe un conjunto de entradas$\{x_1, x_2, \dots, x_D\}$ y devuelve una única salida y. Además, dentro de una RNA existen numerosas conexiones entre las distintas neuronas que la forman. Estas conexiones simulan las conexiones interneuronales del cerebro y, al igual que éstas, pueden establecerse con mayor o menor intensidad. En el caso de las RNA esta intensidad la determinan los `pesos sinápticos` (o simplemente `pesos`). De este modo, cada entrada x_1 de una neurona se encuentra afectada por un peso w_i.

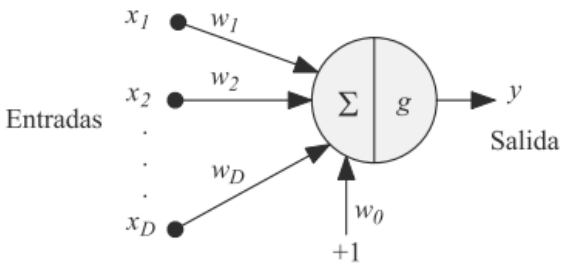

Figura 1 Modelo neuronal (McCulloch y Pitts, 1943).

El primer paso para obtener la salida y de la neurona es calcular la suma ponderada de las entradas, llamada `activación` de la neurona:

$$a = \sum_{i=1}^{D} w_i x_i + w_0$$

Figura 2 Suma ponderada de entradas (Elaboración propia)

donde w_0 es un *umbral* o *sesgo* que se utiliza para compensar la diferencia entre el valor medio de las entradas, sobre todo el conjunto de entrenamiento, y el correspondiente valor medio de las salidas deseadas. Posteriormente, a partir de este valor a se obtiene la salida y de la neurona mediante la aplicación de una función, llamada *función de activación* o de *transferencia* $g(a)$ es decir:

$$y = g(a) = g\left(\sum_{i=1}^{D} w_i x_i + w_0\right) = g\left(\sum_{i=0}^{D} w_i x_i\right)$$

Figura 3 Función de transferencia (Palma & Marín, 2008)

Como se observa, es posible tratar el umbral w0 como un peso más si se supone una entrada añadida x_0 con un valor fijo de 1 (Figura 4). Finalmente, también es posible reescribir esta ecuación en notación vectorial como $g(a) = g(w^T x)$, si tomamos w como el vector de pesos y x como el vector de entradas a la red. La función de transferencia empleada en este modelo básico de McCulloch-Pitts es la función *escalón* definida por la ecuación (Palma & Marín, 2008)

$$g(a) = \begin{cases} 0 & \text{cuando} \quad a < 0 \\ 1 & \text{cuando} \quad a > 0 \end{cases}$$

Figura 4 Función escalón (Elaboración propia)

Red neuronal Monocapa – Perceptrón simple

El *Perceptrón* es la red de neuronas artificiales más sencilla. Está compuesta únicamente por una capa de neuronas de entrada y otra capa de neuronas de salida. Para simplificar, en esta sección se considerará una única salida. Si bien este tipo de red apenas se emplea en la actualidad por las limitaciones que presenta, su estudio es obligado dado que es la base de los métodos actuales.

En 1957 Frank Rosenblatt, basándose en el modelo de McCulloch-Pitts y empleando como regla de aprendizaje una modificación de la propuesta por Hebb, presentó el perceptrón, el primer modelo de red de neuronas artificiales (Rosenblatt, 1962). La arquitectura que Rosenblatt definió para el perceptrón consistía en una primera capa de j neuronas con funciones ϕ_j que se encargaban de transformar los datos de entrada. Estas funciones reciben un subconjunto aleatorio de entradas a través de unos pesos fijos y les aplican

una función de activación de tipo escalón, de nuevo existe un peso especial o sesgo w_0 y, de igual modo se definía una entrada artificial x_0 con valor 1 asociado a este peso, en el caso del perceptrón se considera una función de activación extra $\phi_0 = 1$.

La principal aportación del perceptrón es que la adaptación de sus pesos se realiza teniendo en cuenta el error entre la salida que obtiene la red y la salida que se desearía obtener. Es, por tanto, el primer método de aprendizaje supervisado, en la Figura 5 se muestra la capacidad de este modelo para aprender las funciones lógicas *AND* y *OR*. Para ambas funciones, los pesos w_1 y w_2 tienen idéntico valor, basta variar el valor del umbral (peso w_0) para que dicho modelo represente una función u otra. El valor de `w0` es −1.5 para la función *AND* y −0.5 para la función *OR*.

x_1	x_2	w_1	w_2	$\sum_{i=1}^{2} w_i x_i$	AND	OR
0	0	1	1	0	0	0
0	1	1	1	1	0	1
1	0	1	1	1	0	1
1	1	1	1	2	1	1

Figura 5 Simulación de las funciones AND y OR para el modelo de (McCulloch y Pitts, 1943).

Red neuronal Multicapa – Perceptrón multicapa

Las limitaciones de las redes de una sola capa hicieron que se plantease la necesidad de implementar redes en las que se aumenta el número de capas introduciendo capas intermedias entre la capa de entrada y la capa de salida, de manera que se pudiese implementar cualquier función con el grado de precisión deseado. La función que cumple dicha capa intermedia es tratar de realizar una proyección en la que resulten separables linealmente los patrones de entrada de manera que la unidad de salida pueda realizar una clasificación correcta.

Surge así el perceptrón Multicapa, en adelante *MLP* (MultiLayer Perceptron), cuya arquitectura mostrada en la Figura 6, incluye una o varias capas intermedias de unidades procesadoras, también denominadas capas ocultas porque no tienen conexiones con el exterior.

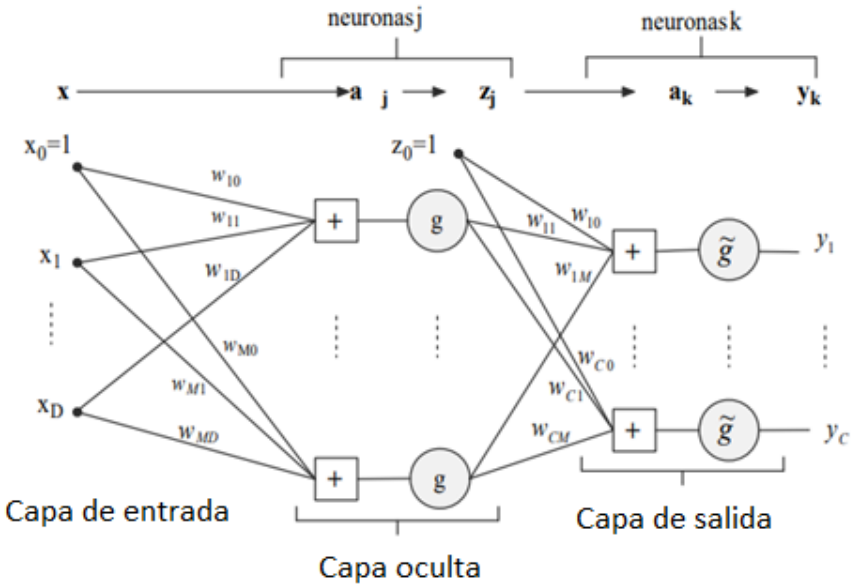

Figura 6 Arquitectura de un perceptrón multicapa con una capa oculta (MultiLayer Perceptron)

Dicho *MLP* consta de *D* entradas, *M* neuronas en su capa oculta y *C* unidades de salida. El nivel de activación *aj* de la neurona *j* de la capa oculta se calcula como una combinación lineal de las *D* entradas x_i que recibe sobre la que, tras aplicar una función de transferencia g se obtiene la salida z_j de dicha neurona:

$$z_j = g(a_j) = g\left(\sum_{i=0}^{D} w_{ji}x_i\right); \quad k = 1, 2, \ldots, M,$$

Figura 7 Función de transferencia g (Elaboración propia)

donde w_{ji}es el peso asociado a la neurona j y la entrada x_i. De manera similar, cada salida de la red se obtiene como una suma ponderada de las salidas de las unidades de la capa oculta, sobre la que se aplica una función de transferencia, es decir, la salida de la neurona k viene dada por:

$$y_k = \tilde{g}(a_k) = \tilde{g}\left(\sum_{j=0}^{M} w_{kj}z_j\right) = \tilde{g}\left(\sum_{j=0}^{M} w_{kj}g\left(\sum_{i=0}^{D} w_{ji}x_i\right)\right); \quad k = 1, 2, \ldots, C.$$

Figura 8 Función ĝ (Elaboración propia)

Es importante destacar que las funciones de transferencia g y ĝ no tienen que ser iguales, por ello la notación que se emplea es diferente. Como veremos posteriormente, la función de transferencia g suele tomar la forma de

las mostradas en la Figura 7, salvo la función escalón, mientras que la forma de la función $\hat{g}$ depende del tipo de problema, Figura 8.

Red neuronal Convolucional (CNN)

Este tipo de redes han marcado una diferencia en la evolución de las redes neuronales, su éxito se debe a la resolución de problemas de visión artificial que, hasta hace poco, se consideraban casi intratables, sirvió para volver a poner de moda las redes neuronales en Inteligencia Artificial. Su explotación comercial dio lugar a lo que hoy entendemos por deep learning. Tal como su propio nombre indica, las redes convolucionales se basan en el uso de convoluciones, una operación matemática familiar para todo el que haya trabajado en procesamiento digital de señales.

Normalmente, se hace referencia a este tipo de redes mediante el acrónimo inglés CNN [Convolutional Neural Network] o, simplemente, con la composición acróstica `ConvNets` o `CNNs (convolutional neural network)`. (Berzal, 2018)

El ejemplo más habitual, una imagen bidimensional que podemos representar mediante una función $f(x,y)$ donde x e y corresponden a las coordenadas espaciales que nos permiten consultar el valor de la imagen en un punto concreto.

Las redes convolucionales son las redes neuronales artificiales que se utilizan habitualmente para resolver múltiples problemas prácticos que requieren procesar imágenes. Por ejemplo, cuando la cámara frontal de un vehículo autónomo capta una señal de tráfico, debe identificar de qué señal concreta se trata (clasificación de imágenes). También puede interesarnos detectar qué tipos de objetos aparecen en la imagen correspondiente a una escena y localizarlos dentro de la imagen (detección de objetos).

Componentes básicos de una red neuronal convolucional (Bagnato , 2018)

- Entrada: Serán los pixeles de la imagen. Serán alto, ancho y profundidad será 1 sólo color o 3 para Red, Green, Blue.
- Capa de Convolución: procesará la salida de neuronas que están conectadas en “regiones locales” de entrada (es decir pixeles cercanos), calculando el producto escalar entre sus pesos (valor de pixel) y una pequeña región a la que están conectados en el volumen de entrada. Aquí usaremos por ejemplo 32 filtros o la cantidad que decidamos y ese será el volumen de salida.

- “CAPA RELU” aplicará la función de activación en los elementos de la matriz.
- POOL ó SUBSAMPLING: Hará una reducción en las dimensiones alto y ancho, pero se mantiene la profundidad.
- CAPA “TRADICIONAL” red de neuronas feedforward que conectará con la última capa de subsampling y finalizará con la cantidad de neuronas que queremos clasificar.

Red neuronal recurrente (RNN)

Por su topología, las redes de tipo feed-forward carecen por completo de memoria. Cuando se añaden conexiones inhibitorias entre neuronas de una misma capa, se consiguen redes cuyo funcionamiento depende del contexto. (Calvo D. , 2017) Si las conexiones entre neuronas de la misma capa se generalizan (o se admiten conexiones de salida de una capa como entradas de una capa anterior), obtenemos redes neuronales con memoria: las redes recurrentes `RNNs (Recurrent Neural Networks)`. Las conexiones de una red recurrente pueden ser dirigidas, como sucede en las redes neuronales multicapas habituales, aunque también pueden ser bidireccionales. En este caso, obtenemos una red que es capaz de completar patrones incompletos, recibidos con ruido u ocultos parcialmente (Berzal, 2018).

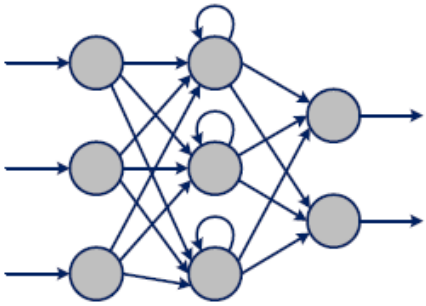

Figura 9 Red neuronal recurrente (Berzal, 2018)

Este tipo de redes recurrentes se caracteriza por su comportamiento dinámico complejo. Ante una señal de entrada ambigua, que se pueda interpretar de distintas formas, una red recurrente puede oscilar entre varios estados semiestables.

Una neurona recurrente transmite la información hacia adelante pero también tiene la característica de enviar la información hacia atrás. Por lo tanto, en cada paso, la neurona recurrente recibe datos de las neuronas anteriores, pero también recibe información de ella misma en el paso anterior.

A efectos prácticos, este tipo de conexiones cíclicas no son eficientes por lo que se establece un despliegue para generar una arquitectura sin ciclos, mucho más adecuado para aplicar las herramientas matemáticas de optimización (Cañadas, 2021).

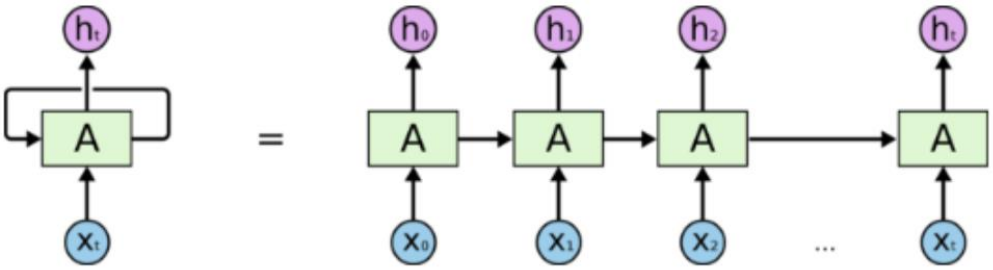

Figura 10 Red neuronal recurrente desenrollada (Borjas, 2021)

Aprendizaje supervisado

El objetivo de cualquier algoritmo de aprendizaje supervisado es construir un modelo de clasificación a partir de un conjunto de datos de entrada, denominado conjunto de entrenamiento, que contiene algunos ejemplos de cada una de las clases que pretendemos modelar además de obtener una descripción precisa para cada clase utilizando para ello los atributos incluidos en el conjunto de entrenamiento. El modelo que se obtiene durante el proceso de aprendizaje puede utilizarse para clasificar nuevos ejemplos (casos cuyas clases se desconozcan) o, simplemente, para comprender mejor los datos de los que disponemos, si nuestro modelo de clasificación es interpretable (algo que no siempre sucede).

Un clasificador tradicional, o regla de clasificación, es una función $f: X \rightarrow Y^*$ definida sobre el conjunto de posibles ejemplos X tal que para todo ejemplo $x \epsilon X$ el clasificador es capaz de asignarle una clase $f(x) = y \epsilon Y^*$

Para construir ese clasificador, utilizaremos un conjunto de ejemplos de entrenamiento de la forma (x, y) en los que x es un vector de características (los atributos que utilizaremos para determinar la clase de cada ejemplo) e $y \epsilon Y$ es la clase a la que pertenece el ejemplo de entrenamiento. Una vez entrenado el clasificador, podremos utilizarlo para tratar de inferir la clase asociada a ejemplos x para los que no conozcamos su clase: $\hat{y} = f(x)$

Aprendizaje no supervisado

En las técnicas de aprendizaje supervisado, estimamos la probabilidad : $\dot{p} = y_k | x$ de que un ejemplo x pertenezca a cada una de las clases de nuestro

problema, las cuales hemos de definir previamente. Nuestro clasificador es, en realidad, un modelo de la distribución de probabilidad condicionada $p(Y \mid X)$ Cuanto mejor sea dicho modelo, más útil nos resultará en la práctica.

En las técnicas de aprendizaje no supervisado, no existen clases predefinidas. Nuestro objetivo será encontrar patrones en los datos de entrada $x \epsilon X$ que nos permitan construir un modelo de la distribución de probabilidad $p(X)$ ¿Qué utilidad puede tener algo así? El aprendizaje no supervisado nos puede servir como herramienta de análisis exploratorio de datos y para pre-procesar los datos antes de utilizar una técnica supervisada (Berzal, 2018).

YoloV5

YOLO es un acrónimo de “You Only Look Once”. Es un modelo muy popular y de alto rendimiento en el campo de detección de objetos, es considerado como la tecnología de punta en detecciones en tiempo real. YOLO-V5 es la quinta generación de los detectores de una sola etapa. YOLO-V5 está implementado en PyTorch. (Montenegro & Flores, 2022)

Marco Práctico

De acuerdo a un estudio realizado por la Organización Mundial de la Salud (OMS) a nivel mundial, señala que la principal causa de daño leve, grave o de muerte en niños es a consecuencia de traumatismo, que se define como “el daño físico que se produce cuando un cuerpo humano se somete brutalmente a algún tipo de energía en cantidades que exceden el umbral de tolerancia fisiológica o cuando se ve privado de unos o más elementos vitales como el oxígeno” (Peden, y otros, 2012). Otras causas comunes que se indican en el reporte de la OMS y que ocasionan daño o lesiones en los niños son: ahogamiento, quemaduras por fuego, intoxicaciones, enfermedades diarreicas, infección de vías respiratorias, malnutrición, enfermedades degenerativas, entre otras, además el 95% de las lesiones en los niños tienen lugar en los países de ingresos bajos y medianos.

“Para alcanzar finalmente el objetivo de desarrollo del milenio de reducir la mortalidad de los niños, es indispensable que adoptemos medidas para abordar las causas de las lesiones en los niños” (Peden, y otros, 2012)

Considerando que los niños tienen una menor estatura en comparación a los adultos, su rango de visión se ve limita o no son visiblemente observables desde ciertos ángulos, por lo que son más propensos a sufrir un accidente.

Otro factor por la que es más probable que una niña o niño sufra un accidente es la capacidad de analizar y reaccionar ante una situación de peligro, que inminentemente es más reducida que el de una persona adulta, el niño aún no tiene plena conciencia lo que representan ciertas acciones o decisiones que puedan dañar su integridad.

Con base al artículo publicado por “Consumidores en Acción” (Andalucia, 2010) los principales escenarios donde se producen los accidentes en el hogar son: la cocina, el cuarto de baño, el dormitorio, la sala de estar y el jardín, siendo el género femenino el más afectado con 57% del total de los casos.

En México de acuerdo a cifras del IMSS (IMSS, 2017), durante periodos vacacionales aumentan hasta un 30% los accidentes de niños en los hogares de las familias mexicanas, los casos que más se reportan en el área de urgencias son: quemaduras, caídas e intoxicación.

En el comunicado de prensa emitido por el INEGI “Estadísticas a propósito del día del niño (30 de abril)” indica que en el 2018 la población de niñas, niños y adolescentes en México representaba alrededor del 30.1% de la población total, esto en cifras significa casi 38.3 millones de personas, a la fecha actual esta población puedo ser aún mayor.

Se estima que más de 25 mil son atendidos en centros hospitalarios del país, de los cuales el 70% ocurren a una distancia a pocos metros de sus padres o tutores (Slim, 2009).

De acuerdo con datos del Consejo Nacional de Población (CONAPO), las entidades que presentan más casos de mortalidad infantil debida a accidentes son Chihuahua, Tlaxcala, Puebla, Guerrero, Oaxaca y Chiapas.

Debido a la gran cantidad de accidentes que ocurren día a día en niños menores, es necesario buscar alternativas o soluciones que ayuden a los padres de familia a tener una constante supervisión de sus hijos.

Existe una serie de medidas que pueden llevarse a cabo para evitar este tipo de eventualidades dañinas para la integridad de las niñas y niños, entre estas posibles soluciones se encuentra el diseñar y desarrollar sistemas inteligentes lo suficientemente autónomos que ayuden a detectar oportunamente eventos que pueda generar algún tipo accidente y que minimice el riesgo.

Este tipo de sistemas puede llevar a cabo varias acciones después de detectar algún evento, que van desde enviar una simple notificación a los

padres de familia, cortar el suministro de energía eléctrica o del gas, activar bloqueos de cerraduras de alacenas, activar robots autónomos para limpiar alguna zona o para recoger objetos en el suelo, activar sistemas de detección de humo y anti-incendios o hasta dar aviso a alguna autoridad de prevención civil o policiaca.

En la actualidad existe una amplia variedad de aplicaciones dedicadas a la seguridad del hogar, pero la mayoría de estos se enfoca en la detección de intrusos; cuando los propietarios del hogar salen de casa, estos sistemas se activan en ciertos horarios, mediante una cámara de video vigilancia detectan algún movimiento o un sonido, es cuando se activa una alarma, se envían notificaciones y se activan sistemas de cerraduras. También en su mayoría se basan en sensores para detectar el humo, estos sistemas son buenos y eficientes, pero muchos no tienen la capacidad de prevenir alguna eventualidad de riesgo, más bien se enfocan de forma reactiva en corregir y dar solución a algún siniestro.

Entre otras funciones de estos sistemas también se encuentra el cumplir órdenes procedentes de alguna fuente, por lo general humana, mediante comandos de voz, estos aparatos reciben instrucciones para activar o desactivar algún aparato inteligente dentro del hogar.

Productos inteligentes como Alexa de Amazon, Siri de Apple y Google Assistant son capaces de procesar comandos de voz e interconectarse con aplicaciones y aparatos para llevar acabo sus tareas asignadas.

De acuerdo con la revista PC Magazine (Colon & Moscaritolo, 2021) en el mercado existe una amplia gama de aparatos inteligentes para diferentes propósitos, que ayudan a mejorar la comodidad y la seguridad en los hogares, solo por mencionar algunos, podemos encontrar:

- Smart Speakers:
    - Amazon Echo Family
    - Google Nest Hub Family
    - Sonos One
- Smart Plugs
    - ConnectSense Smart Outlet 2
    - D-Link mydlink Outdoor Wi-Fi Smart Plug (DSP-W320)
- Home Security Cameras
    - Arlo Ultra
    - Wyze Cam Outdoor
    - Nanit Plus
    - Furbo Dog Camera
- Smart Locks and Home Security Systems

    - August Wi-Fi Smart Lock
    - Vivint Smart Home
    - SimpliSafe Home Security System
- Smart Lighting
    - Yeelight Smart LED Bulb
    - Wyze Bulb

Figura 11 Dispositivos inteligentes dentro del hogar (Haaf, 2020)

La mayoría de estos dispositivos tiene la característica de que incluyen módulos de inteligencia artificial y estos a su vez en algunos casos están basados en redes neuronales, las cuales son un modelo computacional que hace semejanza a las neuronas biológicas (Chaos, 2021).

Retomando un poco el funcionamiento básico de estas redes neuronales, consiste en:

1. La neurona recibe *n* datos de entrada

$$(x_1, x_2, x_3 \dots, x_n)$$

2. Se calculan los pesos ponderados de los valores de entrada

$$(w_1, w_2, w_3 \dots, w_n)$$

3. Se suma un valor adicional (w_0) también llamado *bias*.
4. Estas operaciones vienen dadas por la función lineal

$$(a = w_0 + x_1 w_1 + x_2 w_2 + x_3 w_3 + \cdots + x_n w_n)$$

5. El resultado de estas operaciones viene dado por la función *g(x)* a la que se conoce como *función de activación*. Regularmente

también se le llama como "*binary step*" que devuelve 1 si la variable independiente es igual o mayor que cero y es cero en caso contrario, comúnmente se le llama función escalón.

$$f(x) = \begin{Bmatrix} 0, para\ x < 0 \\ 1, para\ x \geq 0 \end{Bmatrix}$$

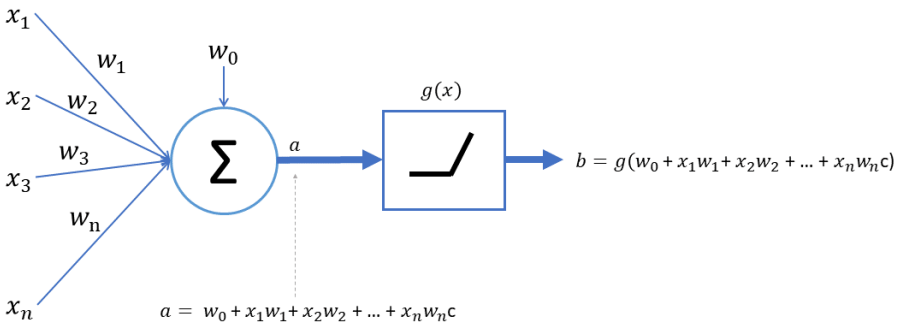

Figura 12 Fórmula detrás de una red neuronal (Chaos, 2021)

Es necesario mencionar que cada uno de estos algoritmos implementa diferentes modelos de machine learning, quizá algunos algoritmos sean mejor que otros para ciertas situaciones y propósitos, sin embargo, lo importante de este trabajo de investigación es determinar cuál es el mejor para obtener los mejores resultados.

YoloV2 es un sistema de código abierto muy eficaz para detectar objetos de videos o imágenes, esto lo hace mediante una red neuronal convolucional, usa Python como lenguaje para programar las redes neuronales, TensorFlow es una liberia de código abierto para el aprendizaje automático fue desarrollado por Google y es capaz de crear y entrenar redes neuronales. En conjuntos con todas estas herramientas, también es sencillo implementar una aplicación que pueda reconocer objetos en videos provenientes de YouTube, archivos de video almacenados y videos provenientes desde una cámara web. (Khandelwal, 2019)

Como se mencionó anteriormente existe una amplia variedad de frameworks especializados para reconocer objetos en videos e imágenes, uno de ellos es Yolo, su versión más reciente y optimizada es la versión 5, fue desarrollada por Ultralytics la misma compañía que desarrollo PyTorch para la versión 3 de Yolo y fue lanzado en junio del 2020, YoloV5 alcanza la misma precisión de la versión 3, pero con 1/4 de complejidad computacional. (Cochard, 2021), YoloV5 está disponible en 4 modelos (*s, m, l, x*) cada uno de ellos ofrece diferente precisión de detección y rendimiento como se observa en la Figura 13.

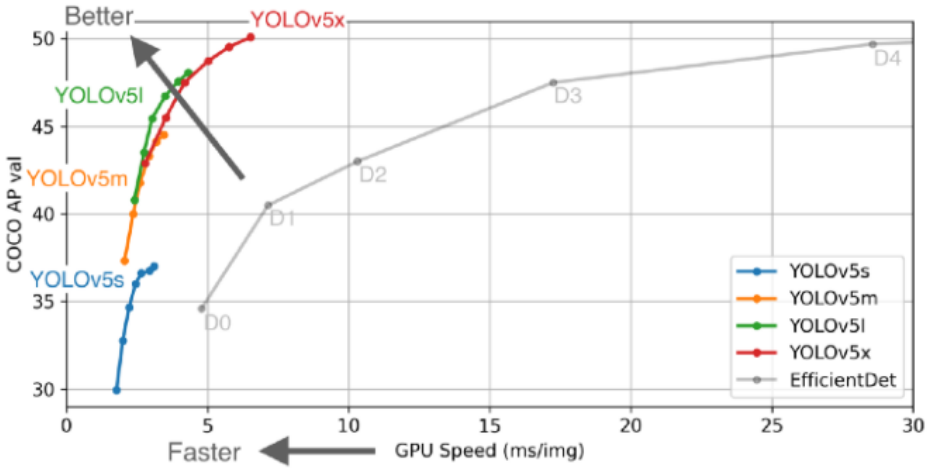

Figura 13 Rendimiento de YoloV5 (Jocher & Koblanski, 2021)

TensorFlow en conjunto con otros algoritmos como DeepFace (algoritmo para detectar rostros en imágenes digitalizadas) People Counting (registro de personas que transitan en una cierta zona), Self Driving Cars (conducción autónoma de automóviles), COCO (Common Objects in Context) es el dataset para entrenar los modelos, son una poderosa combinación para detectar objetos comunes dentro de videos e imágenes en tiempo real. (Keshari, 2020)

En el ámbito nacional y local también se han hecho importantes contribuciones a esta área, principalmente este tipo de aportaciones proceden de estudiantes que quieren obtener algún grado de licenciatura, maestría o doctorado.

Una de estas investigaciones fue realizada en el Instituto Nacional de Astrofísica, cuyo objetivo fue realizar una metodología propia para la identificación de peatones en imágenes aéreas adquiridas desde un vehículo aéreo no tripulado empleando redes neuronales convolucionales, mediante el método explicativo Layer-wise Relevance Propagation junto con la fusión de información obtenida de datos de sensores.

También se emplearon máquinas de soporte vectorial (SVM), arboles de decisión y técnicas de aprendizaje automático. AlexNet y GoogleNet son algunas de las redes neuronales convolucionales implementadas en distintas aplicaciones por su gran capacidad de identificación. Los resultados obtenidos en esta investigación fueron satisfactorios: Las redes neuronales logran identificar en la mayoría de los casos y de forma exitosa la presencia de alguna persona, así mismo, se logra resolver en la mayoría de los casos de forma adecuada la detección de personas.

En los procesos de entrenamiento de las redes neuronales, se decidió no emplear estructuras de redes neuronales pre-entrenadas; esto con la finalidad de generar los mapas de calor que permita visualizar el aprendizaje que cada red neuronal realiza por sí misma, por lo que en esta investigación se optó por entrenar una red neuronal desde cero. (Velázquez J. , 2019)

Otro trabajo importante que se ha hecho, fue el realizar un análisis estructural de los edificios mediante las redes neuronales, en este trabajo se presenta un modelo para realizar un clasificador de imágenes por medio de la aplicación de redes neuronales convolucionales y Teachable Machine, el cual clasifica la carga de imágenes, si una estructura está en condiciones normales o tiene grietas u hoyos.

El programa primero procesa la imagen (lo que se quiere clasificar) y arroja una vista previa de cómo se encuentra la estructura. Posteriormente en forma de vector arroja el resultado de la clasificación. De acuerdo a como se entrenó el modelo, la clasificación es la siguiente [0 = Normal, 1 = Grieta, 2 = Hoyo]. Al utilizar las redes neuronales convolucionales se puede automatizar procesos que la visión humana realiza, algunos ejemplos de esto son: reconocimiento óptico de números y letras, reconocimiento de matrículas, escaneo de códigos, codificación de texto, etc. Gracias a esto se desarrolló un clasificador de imágenes con tres tipos o clases, la cual clasifica con un gran porcentaje de precisión la imagen cargada al modelo previamente entrenado. (Suaste Martínez & Peña Alfaro, 2020)

Entre otras publicaciones que podemos encontrar, está la realizada por un estudiante de la Universidad Nacional Autónoma de México, quien creó un sistema de machine learning entrenado con un conjunto de escenas sintéticas con objetos de interés etiquetados para re-entrenar la red neuronal convolucional YOLOv3. El modelo neuronal obtenido sería utilizado como sistema de detección de objetos en un robot de servicio. Se utilizaron 15 objetos de diferentes clases para medir el desempeño de los sistemas propuestos, los objetos elegidos para las pruebas del sistema fueron: jugo de manzana, tazón azul, lego azul, taza azul, cuchara azul, galletas de chocolate, lego amarillo, taza verde, jugo de piña, galletas de piña, taza morada, plato morado, taza roja y vaso amarillo.

En la mayoría de los archivos de video se obtuvieron resultados superiores al 98% de objetos correctamente segmentados. Sin embargo, objetos como la taza roja y la taza verde tuvieron porcentajes de segmentación menores al 90%, debido a que el sistema falló al detectar los bordes de la base de estos objetos. Se considera que la ejecución del sistema de segmentación de objetos fue exitosa y generó un buen conjunto de imágenes de objetos segmentados para crear las escenas sintéticas. En este trabajo se concluyó

que el módulo de segmentación cumple con el objetivo: desarrollar un sistema de visión que permita segmentar múltiples categorías de objetos utilizando información de una imagen bidimensional adquirida por una cámara digital compacta. (Guzman, 2020)

Entre los trabajos que también se puede destacar está el realizado por un estudiante de la Universidad Autónoma del Estado de México, esta investigación tuvo como objetivo desarrollar un sistema de reconocimiento de objetos para buscar un objeto dentro de un conjunto de objetos de tlapalería almacenados como imágenes digitales de 500 x 500 pixeles.

En este trabajo se utilizaron componente y herramientas de software tales como: Raspberry Pi 2, es una computadora de placa simple de bajo costo desarrollada en Reino Unido, OpenCV es una API de aproximadamente 300 funciones escritas en lenguaje C, está dirigida fundamentalmente a la visión por computadora en tiempo real y Python que, es un lenguaje de programación interpretado, multiplataforma y orientado a objetos.

En dicho trabajo quedo demostrado que una vez que se llevó a cabo la metodología se obtuvieron resultados que muestran la posibilidad de reconocer objetos de una tlapalería, por lo tanto se concluyó que el problema de que una persona no pueda identificar un objeto dentro de una tlapalería podría quedar resuelto si se implementa esta solución. (Gonzalez Osorio, 2017)

Finalmente cabe mencionar que se realizó un trabajo de investigación también en la Universidad Autónoma del estado de México, donde se implementó un sistema de reconocimiento facial, para la detección de usuarios autorizados para el ingreso y conducción de un automóvil, se utilizaron las siguientes herramientas: algoritmos eigenfaces, análisis de componentes principales e histogramas de patrones locales binarios, Raspberry Pi 3 y GPS. En el experimento de la fase de entrenamiento se crearon las etiquetas que sirven para complementar la fase de reconocimiento facial. Al usar una cámara de teléfono celular, puesta dentro de un automóvil, reconoció al usuario asignándole la etiqueta respectiva.

Por lo que se considera como verdadero positivo. Si el usuario no estuviera en la base de datos, el sistema lo detectara como desconocido. Después de evaluar el rendimiento de los algoritmos con la matriz de confusión, cumple con un 85% de la clasificación correcta de los resultados, por lo que satisface un porcentaje aceptable en el reconocimiento, pues un 3% genera falsos negativos, causando también que se envíen las alertas. Muchos de los resultados se vieron afectados por la iluminación, el brillo, el contraste y la textura de las imágenes. (Mateo Jiménez, 2020).

Capítulo III. Metodología

Se realizó una profunda investigación y revisión de varias publicaciones con respecto al tema de inteligencia artificial y visión por computadora, muchas de estas investigaciones están enfocadas en el reconocimiento de objetos en vías públicas como el ya reconocido proyecto y lucrativo automóvil autónomo Tesla, que se enfoca en el reconocimiento de objetos de la vía pública e identifica obstáculos que pudiera obstruir su camino, otro ejemplo es la implementación de sistemas de reconocimientos facial, estos frecuentemente se instalan en los aeropuertos, en medios de transporte como estaciones de tren, terminal de autobuses, en estadios deportivos y lugares donde es común la concentración de personas, esto con la finalidad de garantizar la seguridad de las personas, la mayoría de los algoritmos de machine learning están entrenados con objetos podemos encontrar en la vía pública, debido a que no es necesario solicitar o tramitar algún permiso para almacenar y procesar esta información.

Figura 14 Detección de objetos en vía pública (Meel, 2021)

Para complementar la información teórica se encontraron varios documentos muy interesantes con respecto al tema de inteligencia artificial aplicado a la visión por computadora.

La recolección y etiquetado de los datos, para este caso imágenes, es una de las etapas más importantes para cumplir con los objetivos planteados en el proyecto, cuyo propósito es elaborar un dataset de imágenes que posteriormente se van a utilizar para el entrenamiento de nuestro modelo que será el encargado de realizar las detecciones de eventualidades.

Actualmente ya existen dataset muy completos que nos facilitan esta tarea de recolección, sin embargo, para nuestro propósito es necesario generar un nuevo dataset personalizado, que cumpla con el objetivo de reconocer objetos y eventos que representan un riesgo para los niños. Para esto se requiere de tomar una serie de fotografías de objetos como, por ejemplo, contactos de luz, juguetes tirados en el suelo, objetos de cocina, bebidas alcohólicas, medicamentos, etc. Con estas imágenes se pueda entrenar el modelo y generar los pesos necesarios para que la detección de objetos pueda tener resultados precisos y disminuir los falsos positivos.

Otro método es la recolección de imágenes en la web, mediante la técnica scraping web, se puede extraer una gran cantidad de imágenes desde la web, partiendo de palabras claves, se puede extraer imágenes, almacenarlas y procesarlas para entrenar el modelo machine learning.

Un tema fundamental para llevar a cabo este proyecto, es la visión por computadora hoy en día hay una gran cantidad de algoritmos y métodos para procesar imágenes y video. Con base al artículo publicado por Appsilon (Świeżewsk, 2020) el algoritmo de YOLO (Redmon, Joseph, Farhadi, & Ali, 2018) es uno de los más utilizados para detectar objetos en las imágenes y videos, una de las principales ventajas de este algoritmo es que no se tiene que programar el modelo de machine learning desde cero, sino que ya cuenta con modelos entrenados que se pueden utilizar e implementar. Una de las implementaciones de YOLO es Darknet su principal característica es que está enfocado para construir aplicaciones en tiempo real, gracias a que está desarrollado en el lenguaje de programación C y CUDA technology, esta optimizado para trabajar con GPU.

En el foro de la revista Quora (Foro, 2017) se mencionan algunos otros algoritmos importantes como por ejemplo:

- SSD (Single Shot Detector)
- Fast R-CNN
- Faster R-CNN
- R-CNN (Region-based Convolutional Neural Networks)
- R-FCN (Region-based Fully Convolutional Network)

Muchos de estos algoritmos se basan en redes neuronales y su funcionamiento es muy básico y muy similar al funcionamiento de una red neuronal biológica. La fórmula básica de una red neuronal se puede observar en la Figura 15.

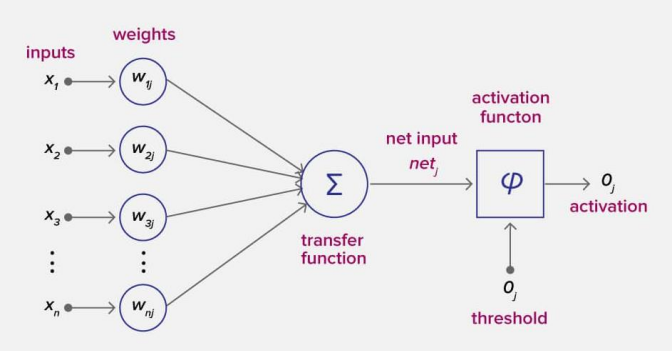

Figura 15 Formula detrás de una red neuronal (Calvo J. , 2020)

Existen una extensa variedad de publicaciones y muchas de estas están enfocadas a la detección de objetos en la vía pública, YoloV5 se especializa en la detección de objetos comunes que se pueden encontrar en la vía pública. Figura 16.

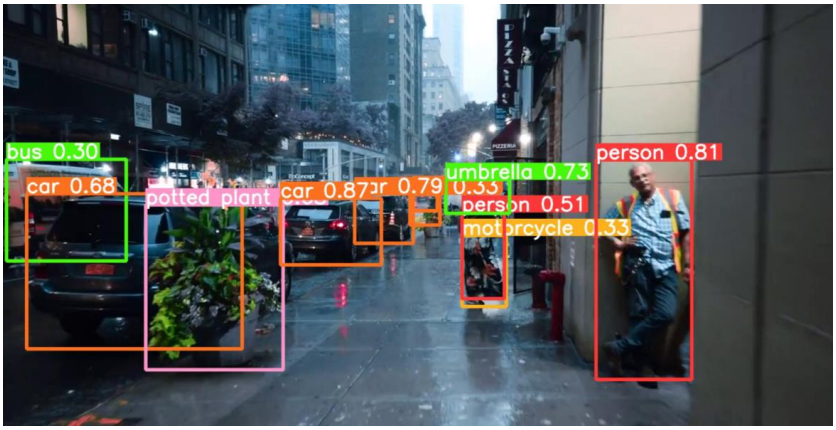

Figura 16 Detección de objetos en la vía pública (Elaboración propia)

A nivel mundial encontramos que la gran mayoría de estos temas ya están muy avanzados y desarrollados, sobre todo en la parte del continente asiático, en países como China, India, Corea del Sur, Japón, ya cuentan con sistemas que están a la vanguardia con lo último en tecnología, en especial con la inteligencia artificial, esto se debe principalmente a la educación que reciben los estudiantes asiáticos, muchos de estos se forman en universidades de Estados Unidos (Alatorre, 2021) esto ha contribuido a que el avance tecnológico de los países asiáticos se haya acelerado notablemente en los últimos años y sin duda en un futuro próximo, Estados Unidos no será la principal potencia tecnológica a nivel mundial.

Los frameworks cuyo propósito es el reconocimiento de objetos en las imágenes, entre los que podemos encontrar PyTorch, este framework puede ser utilizado para detectar objetos en videos de YouTube, como por ejemplo se puede colocar un video de automóviles circulando en una carretera y este framework será capaz de reconocer dichos automóviles.

Este framework se base en modelos de machine learning pre-entrenados para detectar objetos y se apoya del uso de tarjetas de video para mejorar el rendimiento de la aplicación, usa la librería de OpenCV para dibujar el contorno de los objetos detectados. Al detectar un objeto coloca la etiqueta de lo que detecto, por ejemplo, si detecto un automóvil el sistema colocará en el contorno la etiqueta “car”. Este framework facilita la implementación de un modelo de machine learning para detectar objetos de cualquier video de YouTube de forma sencilla. (Clemens W. , 27)

Capítulo IV. Desarrollo

Para elaborar el sistema de detección de riesgos en los hogares se desarrolló un sistema con base a las siguientes etapas y de las cuales se mencionan a continuación Figura 17.

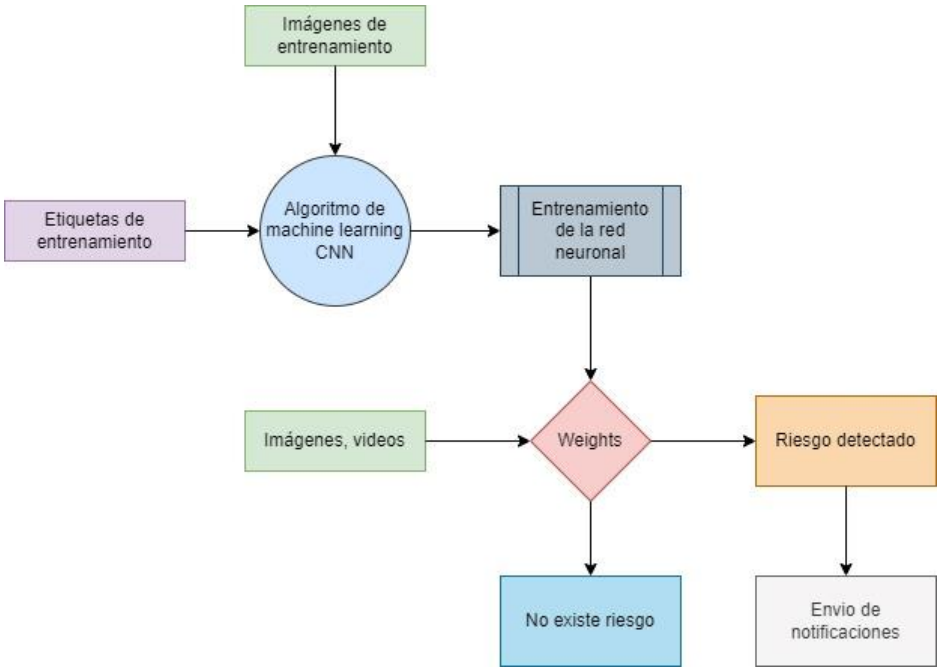

Figura 17 Fases de desarrollo del sistema (Elaboración propia)

- *Creación del dataset, elección de imágenes (niños, objetos que representen un riesgo para los niños).*

  Esta etapa es fundamental para el correcto funcionamiento del sistema, cuyo propósito es encontrar imágenes adecuadas para realizar el entrenamiento de las redes neuronales y así obtener resultados de acuerdo a las estimaciones de las detecciones que se pretenden identificar en los videos e imágenes a analizar para la detección de riesgos, en primera instancia se realizó una búsqueda de imágenes en la web, mejor conocido como web scraping (raspado web en español) para cumplir con este propósito se desarrollaron herramientas mediante el lenguaje de programación python con la finalidad de hacer una búsqueda en la web de imágenes publicas, por medio de palabras claves, es decir, existen librerías tales como las que provee google o Bing (buscador de Microsoft), para realizar este tipo de búsquedas. Figura 18.

```
1  from bing_image_downloader import downloader
2  downloader.download('cuchillo', limit=100,  output_dir='dataset',
3  adult_filter_off=True, force_replace=False, timeout=15)
4
```

Figura 18 Ejemplo de código web scraping (Elaboración propia)

Un tema fundamental es la privacidad de la información, para cumplir con el propósito de este trabajo de investigación se utilizaron imágenes de niños en sus hogares realizando actividades comunes, como jugar, caminar o conviviendo con sus padres, por ello se realizado la búsqueda de imágenes públicas que están al alcance de todos y que de alguna manera con el consentimiento de los padres, las imágenes o los videos fueron publicados en la web.

Además de niños, también se realizó la búsqueda de imágenes de personas adultas, esto con la finalidad de poder identificar entre un adulto y un niño, la idea principal del sistema no es solo identificar niños en riesgo, sino que también es tener la capacidad de identificar entre un niño y un adulto, aunque de primera impresión suena bien identificar cualquier persona en riesgo, por el momento solo se delimito el proyecto a identificar niños. Entre los objetos que se realizó la búsqueda fueron contactos de luz, cuchillos, herramientas (desarmadores, martillos, etc.).

Se realizó la búsqueda de aproximadamente 1,700 imágenes en total, para el entrenamiento. Figura 19.

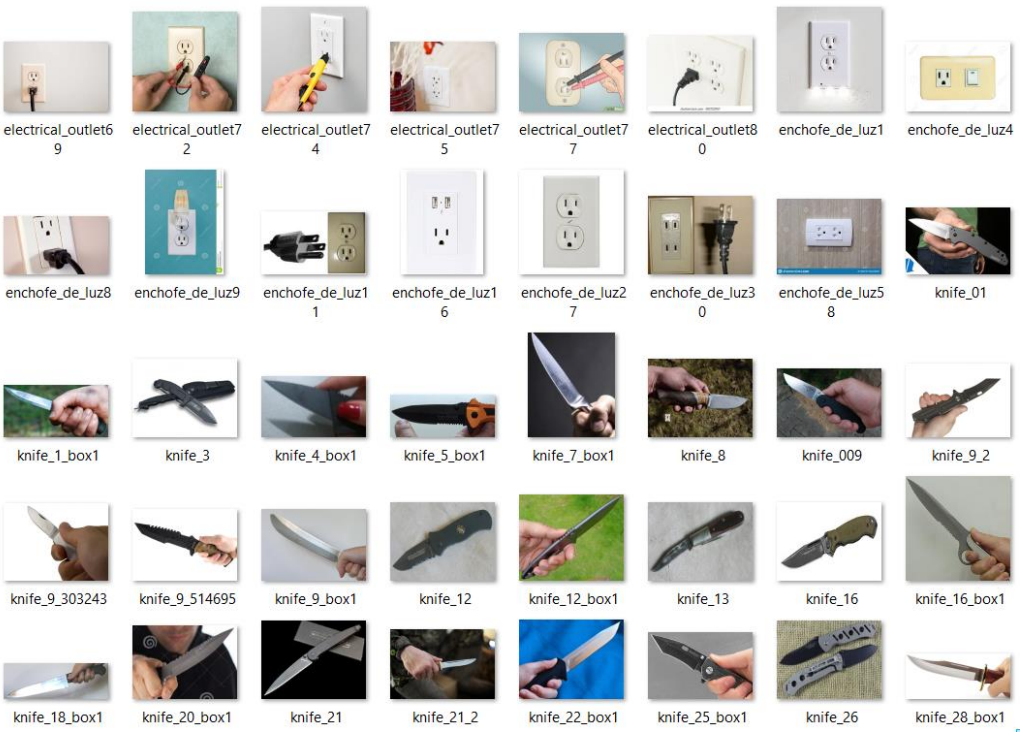

Figura 19 Dataset para el entrenamiento (Elaboración propia)

Para crear un dataset lo más completo y útil posible, es necesario tener la mayor cantidad de imágenes con características similares, como se puede asumir existe una gran variedad de imágenes en la web, diferentes tamaños, colores, formas, situaciones, etc. Por lo cual es recomendable concentrar la mayor cantidad de imágenes con caracteriscas en común para incrementar la precisión en el entrenamiento.

- *Etiquetado de las imágenes, indicar dentro de la imagen mediante un cuadro (bounding box) donde se encuentra el niño o los objetos de riesgo.*

Una vez que se tiene el dataset con las imágenes adecuadas, el siguiente paso es crear el bounding box dentro de las imágenes, la herramienta que se utilizó fue `labelImg v1.8.6` Figura 20 esto es necesario ya que al utilizar YoloV5 se requieren dos tipos de archivos para el entrenamiento, un archivo .jpg y un archivo .txt, explicados a continuación.

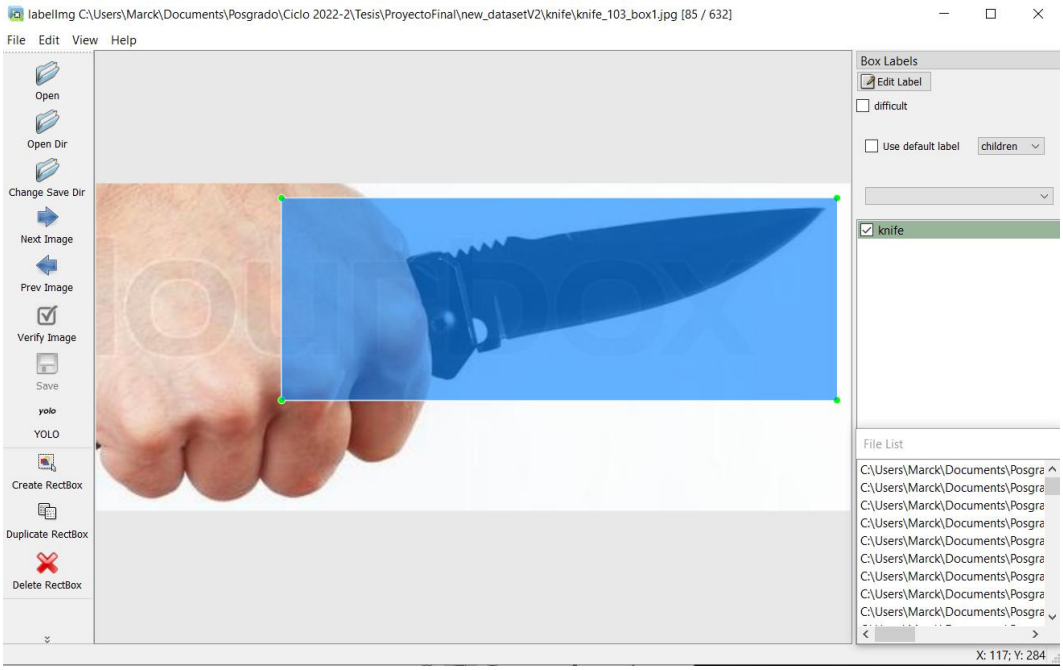

Figura 20 Bounding box para detectar cuchillos (Elaboración propia)

- Labels - archivo .txt contiene las coordenadas del bounding box del objeto a identificar, el formato consta de lo siguiente, Figura 21:

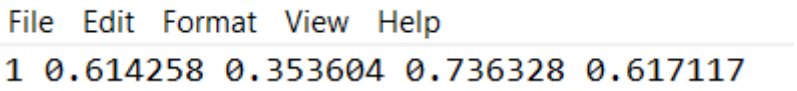

Figura 21 Estructura del archivo label (Elaboración propia)

  - clase del objeto
  - coordenada x
  - coordenada y
  - ancho
  - alto

- Images - son las imágenes que se obtuvieron de la web, este archivo debe tener el mismo nombre que el archivo .txt del label Figura 22

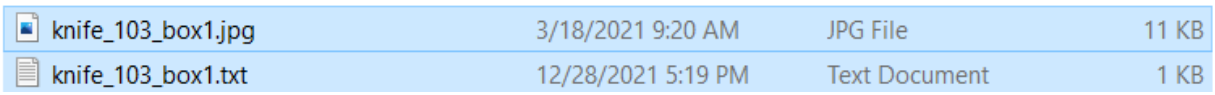

Figura 22 Archivos generados por el proceso de bounding box (Elaboración propia)

Este proceso se realiza con cada una de las imágenes que se obtuvieron desde la web, cabe mencionar que este es un proceso tedioso y tardado debido a que se tiene que hacer manualmente con cada una de las imágenes.

➢ *Elegir la red neuronal y el modelo para realizar el entrenamiento.*

La arquitectura de la red neuronal de YoloV5, está compuesta básicamente por tres modelos Figura 23.

- Model Backbone - utiliza la red neuronal convolucional CSPDarknet (Cross Stage Partial Network) para la detección de objetos, su propósito es dividir el mapa de características en dos para después fusionarlas mediante una jerarquía entre etapas.
- Model Neck - basada en PANet (Path Aggregation Network) y SSP (Spatial Pyramid Pooling) trata sobre la segmentación semántica para la clasificación de los pixeles de una imagen para determinar y darle un sentido a qué clase de objeto se refiere. Una de las razones principales de elegir este tipo de redes es por la habilidad de conservar la información espacial con precisión por consecuencia ayuda a la localización de pixeles. (opengenus, 2022)
- Model Head – se encarga de realizar las operaciones finales presenta el resultado final, determina la clase a la que pertenece el objeto y crea cuadros delimitadores (bounding boxes) en los objetos detectados.

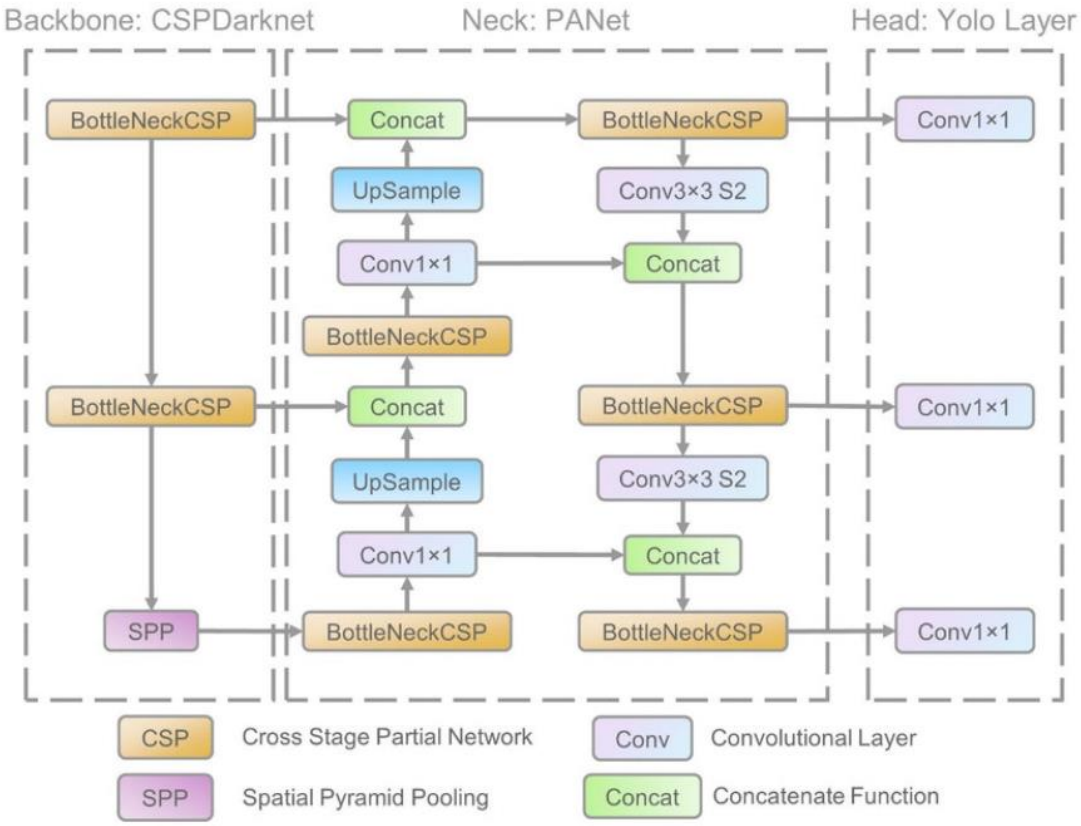

Figura 23 Arquitectura de YoloV5 (opengenus, 2022)

Para implementar el sistema de detección de riesgo se utilizó el modelo YoloV5l, cuyas características se encuentran en la Figura 24:

Model	size (pixels)	mAP^{val} 0.5:0.95	mAP^{val} 0.5	Speed CPU b1 (ms)	Speed V100 b1 (ms)	Speed V100 b32 (ms)	params (M)	FLOPs @640 (B)
YOLOv5n	640	28.0	45.7	**45**	**6.3**	**0.6**	**1.9**	**4.5**
YOLOv5s	640	37.4	56.8	98	6.4	0.9	7.2	16.5
YOLOv5m	640	45.4	64.1	224	8.2	1.7	21.2	49.0
YOLOv5l	640	49.0	67.3	430	10.1	2.7	46.5	109.1
YOLOv5x	640	50.7	68.9	766	12.1	4.8	86.7	205.7

Figura 24 Características del modelo YoloV5l (Ultralytics, 2022)

- *Entrenamiento de la red neuronal*

Antes de comenzar el entrenamiento de la red neuronal hay que crear la estructura de directorios del proyecto, para ello se puede obtener una copia del proyecto junto con su estructura de directorios del siguiente link:

https://github.com/ultralytics/yolov5

Al clonar el proyecto desde GitHub en la máquina local se genera una estructura similar a la Figura 25.

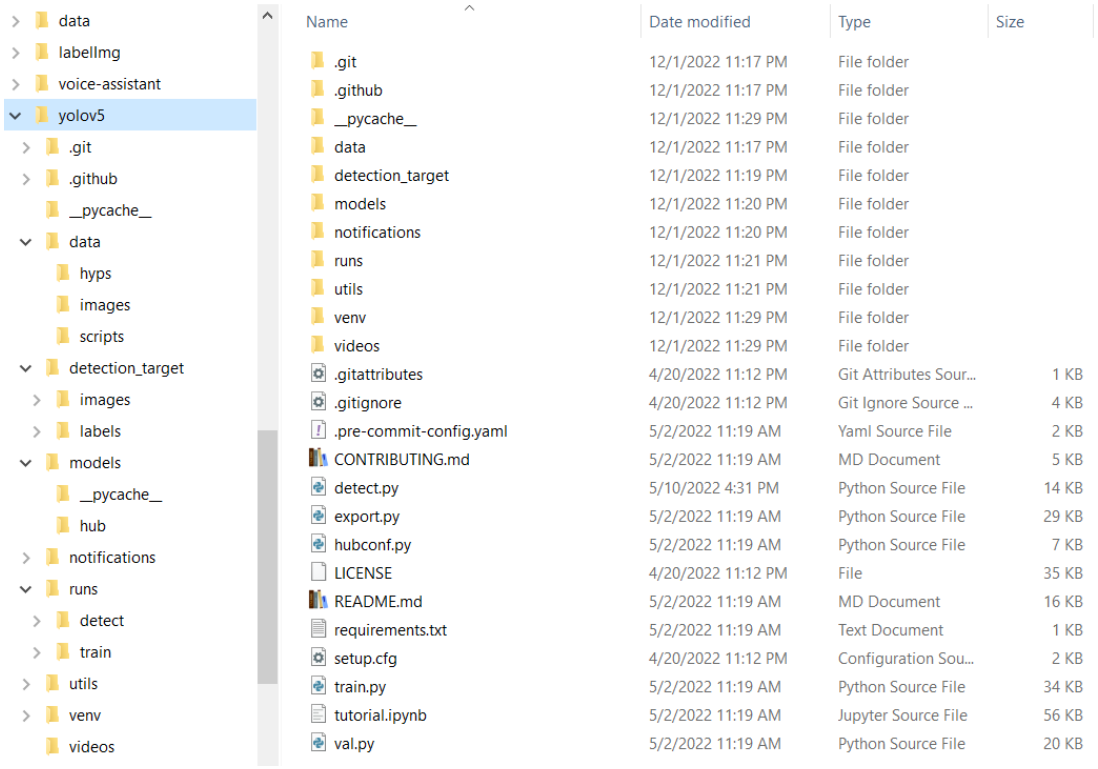

Figura 25 Estructura de directorios de yolov5 (Elaboración propia)

Para realizar el entrenamiento personalizado de la red neuronal en primera instancia se elige una ubicación dentro del directorio del proyecto para almacenar las imágenes y los labels, para este caso se eligieron las siguientes ubicaciones

- Imágenes obtenidas de la web para el entrenamiento - `/yolov5/detection_target/images`
- Labels generados para realizar el entrenamiento - `/yolov5/detection_target/labels`

Otro factor importante para realizar el entrenamiento fue la configuración de los archivos:

- `/yolov5/data/detection_target.yaml` - se especifica la ubicación de los directorios donde se encuentran las imágenes para el entrenamiento, pruebas y validación. Además, se especifica el número de clases y el nombre las clases, también pueden ser considerados como los objetos a detectar, el contenido del archivo se puede observar en la Figura 26.

Clases para el entrenamiento:

```
names:
["children","outlet","knife","tools","adult"]
```

```
train: ./detection_target/images/train/
val:  ./detection_target/images/val/
test: ./detection_target/images/test/

# number of classes
nc: 5

# class names
names: ["children","outlet","knife","tools","adult"]
```

Figura 26 Contenido del archivo detection_target.yaml (Elaboración propia)

- `/yolov5/models/yolov5l_detection_target.yaml` – en este archivo se indica la configuración necesaria para el Backbone y Head, mencionados anteriormente. El contenido del archivo se puede ver en la Figura 27

```yaml
# Parameters
nc: 5  # number of classes
depth_multiple: 1.0  # model depth multiple
width_multiple: 1.0  # layer channel multiple
anchors:
  - [10,13, 16,30, 33,23]  # P3/8
  - [30,61, 62,45, 59,119]  # P4/16
  - [116,90, 156,198, 373,326]  # P5/32
# YOLOv5 v6.0 backbone
backbone:
  # [from, number, module, args]
  [[-1, 1, Conv, [64, 6, 2, 2]],  # 0-P1/2
   [-1, 1, Conv, [128, 3, 2]],  # 1-P2/4
   [-1, 3, C3, [128]],
   [-1, 1, Conv, [256, 3, 2]],  # 3-P3/8
   [-1, 6, C3, [256]],
   [-1, 1, Conv, [512, 3, 2]],  # 5-P4/16
   [-1, 9, C3, [512]],
   [-1, 1, Conv, [1024, 3, 2]],  # 7-P5/32
   [-1, 3, C3, [1024]],
   [-1, 1, SPPF, [1024, 5]],  # 9
  ]
# YOLOv5 v6.0 head
head:
  [[-1, 1, Conv, [512, 1, 1]],
   [-1, 1, nn.Upsample, [None, 2, 'nearest']],
   [[-1, 6], 1, Concat, [1]],  # cat backbone P4
   [-1, 3, C3, [512, False]],  # 13
   [-1, 1, Conv, [256, 1, 1]],
   [-1, 1, nn.Upsample, [None, 2, 'nearest']],
   [[-1, 4], 1, Concat, [1]],  # cat backbone P3
   [-1, 3, C3, [256, False]],  # 17 (P3/8-small)
   [-1, 1, Conv, [256, 3, 2]],
   [[-1, 14], 1, Concat, [1]],  # cat head P4
   [-1, 3, C3, [512, False]],  # 20 (P4/16-medium)
   [-1, 1, Conv, [512, 3, 2]],
```

Figura 27 Contenido del archivo yolov5l_detection_target.yaml (Elaboración propia)

Una vez que se tienen los archivos necesarios, es momento de comenzar con el entrenamiento, para ello se configuraron los siguientes parámetros para la ejecución del script:

- 30 epochs
- 2 batch
- 16 workers

- ~ 1,700 imágenes.

```
python train.py --batch 2 –cfg
yolov5l_detection_target.yaml --weights '' --data
detection_target.yaml --img 640 --epochs 30 --hyp
hyp.scratch-med.yaml --workers 16 --name
yolo_detections
```

El entrenamiento se realizó en una computadora portátil con las siguientes características:

- Procesador Intel Core i7
- Memoria RAM 8GB
- Tarjeta de video NVIDIA GeForce GTX 970M 3.025 GB
- Sistema operativo Linux Ubuntu 22.04

Al ejecutar el comando, el entrenamiento comienza, en primera instancia se recopila todas las configuraciones que se realizan en los archivos mencionados anteriormente, también se detecta si existe alguna tarjeta de video (unidad de GPU) para ejecutar el entrenamiento sobre dicho dispositivo, en caso de no encontrar ningún GPU configurado, se realiza el entrenamiento directamente sobre el CPU, como se puede ver en la Figura 28.

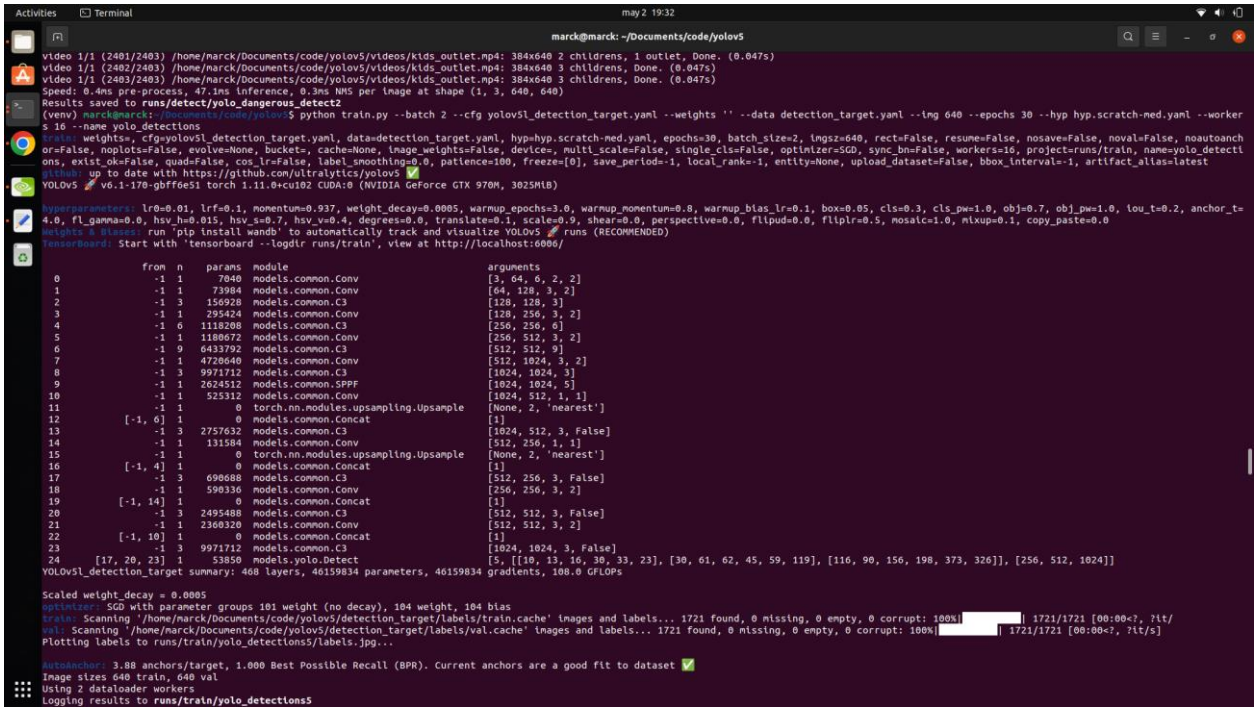

Figura 28 Entrenamiento del modelo (Elaboración propia)

El entrenamiento tardo en total 5.3 horas, con lo cual se obtuvieron los pesos, la matriz de confusión, curvas de precisión y labels.

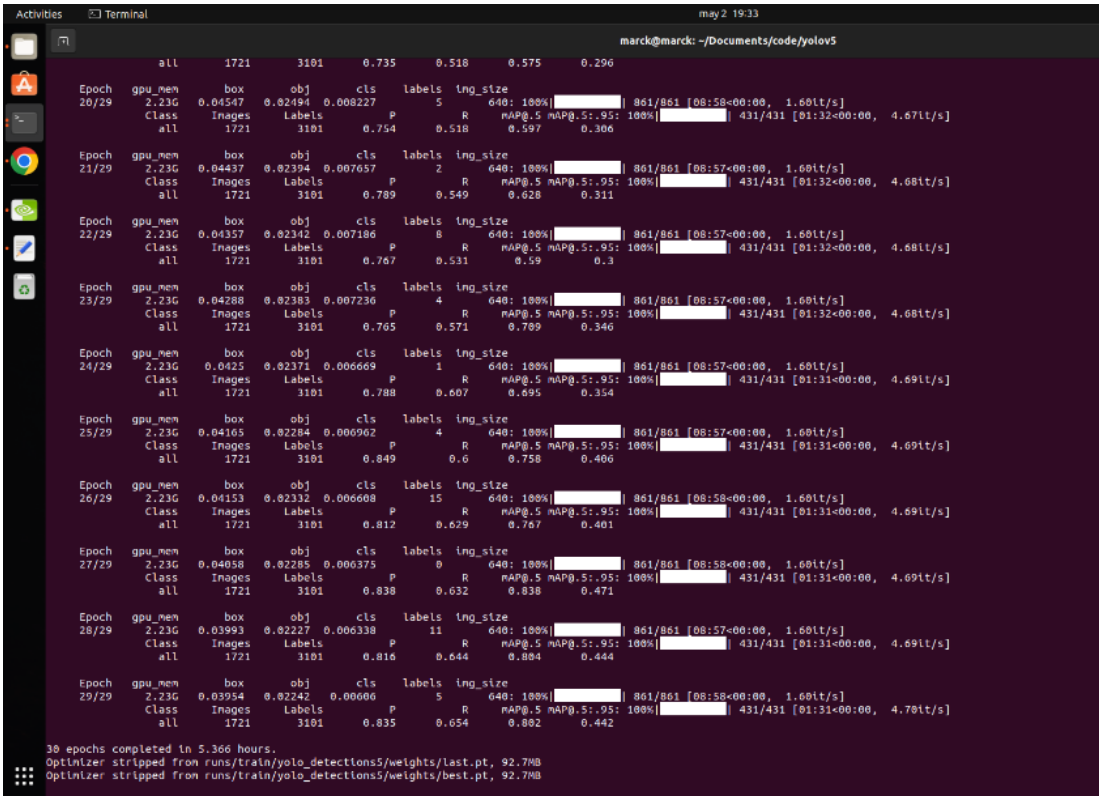

Figura 29 Entrenamiento con 30 épocas (Elaboración propia)

Para visualizar el desempeño del clasificador se utiliza la matriz de confusión donde se puede observar que los mejores resultados para los positivos verdaderos (True Positives) fue para la detección de niños (children) 0.98, contacto de luz (outlet) 0.93, cuchillo (knife) 0.84 y por último la clase de herramientas (tools) 0.71, en caso contrario, donde se obtuvieron los peores resultados y prácticamente casi nula fue en la clase de adultos (adult) donde es necesario volver a reentrenar el modelo con un dataset más numeroso y completo sobre adultos, solo para el caso que se requiera detectar esta clase, pero como no es el propósito de esta investigación detectar adultos, no va ser necesario volver hacer el entrenamiento, Figura 30.

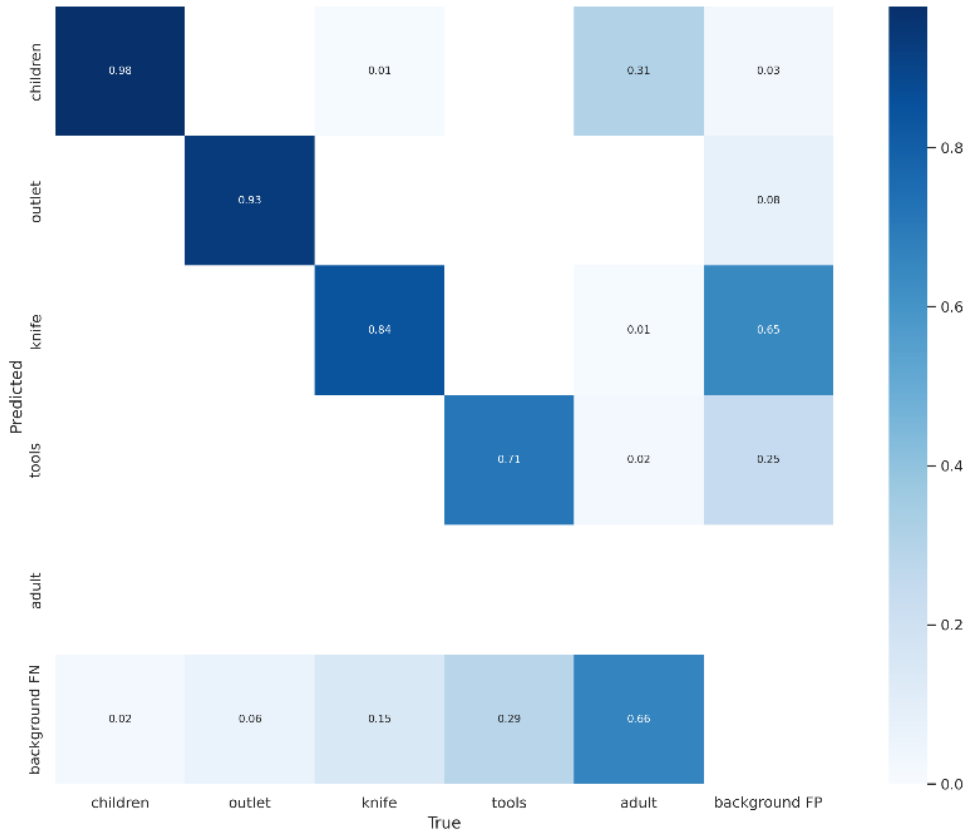

Figura 30 Matriz de confusión (Elaboración propia)

En la siguiente imagen se puede observar los porcentajes de precisión y confidencialidad que se obtuvieron con el entrenamiento, para la mayoría de las clases se obtuvieron resultados satisfactorios, con una confidencialidad cercana al 80%, excepto para la clase adulto, donde se observa que no cubre ni el 10% en la confidencialidad, Figura 31.

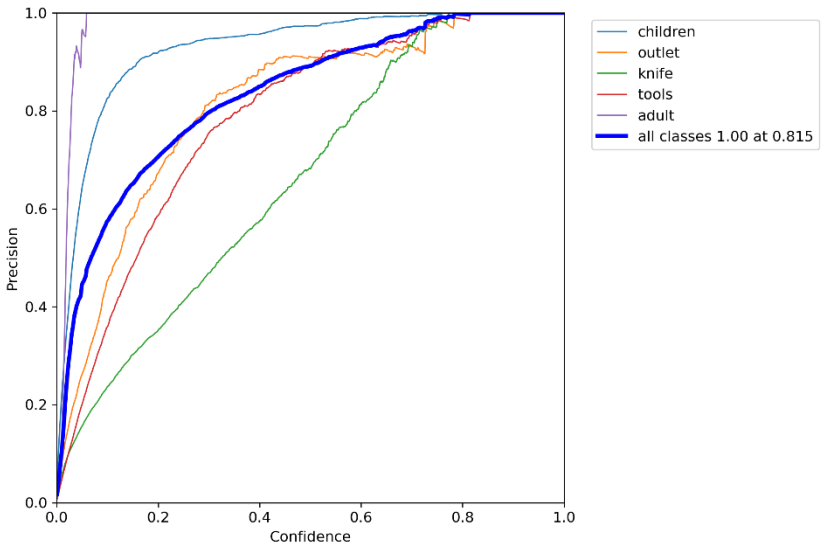

Figura 31 Curva de precisión y confidencialidad (Elaboración propia)

- *Generar los pesos (weights).*

Cuando finaliza el proceso de entrenamiento con las imágenes, los labels y el modelo seleccionado, se generan los pesos (weights) en la carpeta `/yolov5/runs/train/yolo_detections[#]/weights`

- `best.pt` - es el peso con los mejores resultados de todas las `epochs` del entrenamiento
- `last.pt` - es el peso del resultado del último `epoch` del entrenamiento

Son útiles para el momento en que se realiza la detección de objetos, son un parámetro fundamental que determina de acuerdo a las características del objeto a detectar, si cumple con los umbrales establecidos de acuerdo a nuestro modelo elegido y el entrenamiento si el objeto cumple con las características de alguna clase.

- *Procesamiento de videos e imágenes para la detección de riesgos.*

La idea principal del proyecto es detectar en tiempo real alguna situación de riesgo que pueda presentar algún niño menor de 12 años, por lo que para hacer el análisis el sistema puede leer de varias fuentes como, por ejemplo, analizar una imagen, un video, desde una cámara web, o desde un video streaming, entre otros. En la Figura 32 se observan todas las opciones que se disponen para hacer el análisis de videos o imágenes.

```
python detect.py --weights yolov5s.pt --source 0                               # webcam
                                               img.jpg                         # image
                                               vid.mp4                         # video
                                               screen                          # screenshot
                                               path/                           # directory
                                               list.txt                        # list of images
                                               list.streams                    # list of streams
                                               'path/*.jpg'                    # glob
                                               'https://youtu.be/Zgi9g1ksQHc'  # YouTube
                                               'rtsp://example.com/media.mp4'  # RTSP, RTMP, HTTP stream
```

Figura 32 Fuentes disponibles en Yolov5 (Ultralytics, 2022)

Para fines demostrativos se utilizaron videos de YouTube para probar las detecciones en la aplicación, el comando que se utilizo fue el siguiente:

```
python detect.py --source [url del video youtube] --weights
runs/train/yolo_detections5/weights/best.pt --conf
0.70 --name yolo_dangerous_detect
```

Donde los parámetros importantes a explicar son los siguientes:

- `source` - se indica la fuente desde donde se van hacer las detecciones
- `weights` - se indica los pesos del entrenamiento que resulto con mejores validaciones y detecciones durante el entrenamiento
- `conf` - es el porcentaje de confiabilidad que se tiene que el objeto analizado sea el que se quiere detectar, para este caso solo se mostraran los objetos que tengan un porcentaje de confiabilidad mayor al 70%
- `name` - prefijo del nombre de la carpeta donde se almacenarán las detecciones

- *Validar la existencia de algún riesgo dentro del video o de las imágenes y envió de notificaciones*

  El propósito de este sistema es detectar en tiempo real algún riesgo atreves del monitoreo con video cámaras dentro del hogar por lo que se sugiere que el input del sistema o fuente sea un video de streaming, como ya se mencionó anteriormente el sistema soporta varios tipos de fuente entre ellos el video streaming, para ello es necesario adquirir y configurar equipo especializado que tenga la capacidad de transmitir por medio de protocolos como, por ejemplo: RTSP, RTMP, HTTP o HTTPS Stream, por seguridad, si se decide transmitir el video por estos canales, es indispensable y obligatorio cifrar la señal para evitar algún tipo de intrusión o hackeo a datos sensibles con los que se va estar trabajando y resguardar la privacidad de la información.

  Para fines demostrativos como ya se mencionó anteriormente se utilizaron videos públicos de youtube para el análisis de detección de riesgos, cuando se ejecuta la detección de objetos sobre el video, se puede obtener imágenes como las siguientes, Figura 33.

Figura 33 Detección de niño y herramientas (Elaboración propia)

En la figura 33 se puede observar que se detectan objetos que fueron clasificados y entrenados para que sean identificados como herramientas, a su vez también se hizo la clasificación y el entrenamiento de niños para que estos se puedan identificar dentro de los videos. Aquí es importante mencionar que para obtener un resultado satisfactorio en el entrenamiento y en la detección de niños, como es sabido existe una gran variedad de niños por lo que se debe hacer un entrenamiento sobre niños que compartan ciertas caracteriscas físicas, para evitar que falle nuestro entrenamiento al no encontrar caracteriscas comunes entre las imágenes de los niños que queremos identificar. Como fue el caso para identificar personas adultas, para nuestro entrenamiento no se colocó el suficiente número de imágenes de adultos y cuando se hizo el entrenamiento no se encontraron características en común con las imágenes que se colocaron por lo que el entrenamiento para la clase adulta resulto fallido como se puede observar en la anterior Figura 31.

Lo importante de incluir imágenes de personas adultas en el entrenamiento, es que el sistema es capaz de identificar un niño de una persona adulta, aun que la persona adulta no es identifica dentro de un bounding box, la imagen del niño si se identifica correctamente, como se aprecia en la Figura 34.

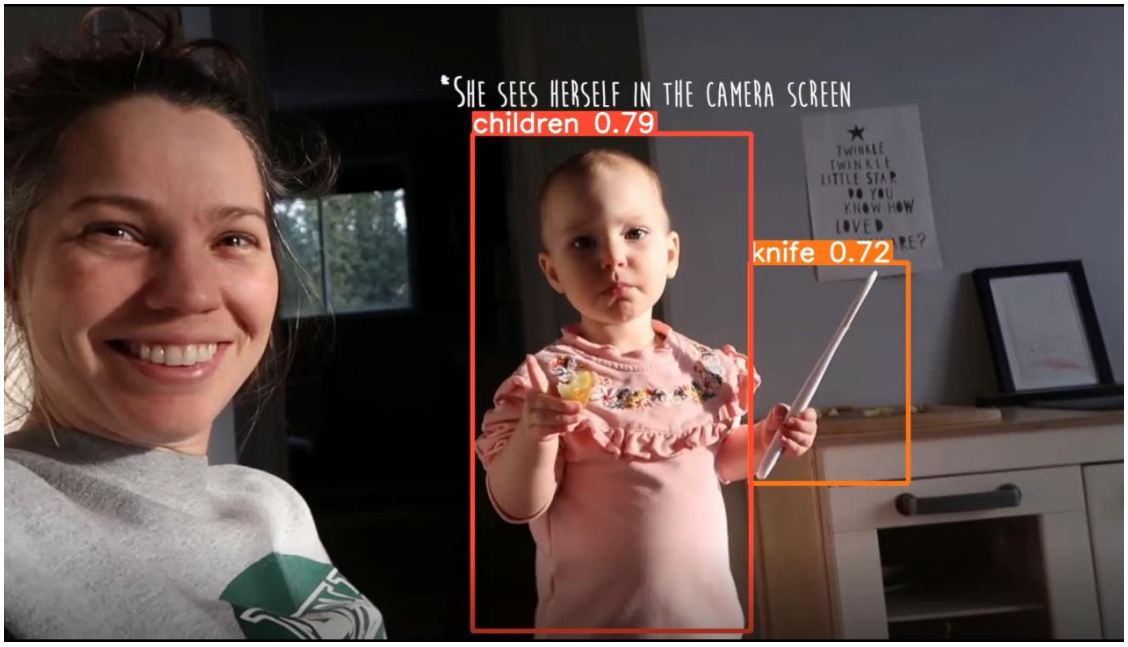

Figura 34 Identificación de niño, adulto descartado (Elaboración propia)

Una vez que se cuenta con las herramientas necesarias para hacer las detecciones, se puede comenzar a enviar las notificaciones si coincide que dentro de un mismo frame existe un niño y un objeto que pueda representar un riesgo para el infante, si cumple esta condición, el sistema comienza a enviar notificaciones por medio de correos electrónicos de los padres o tutores del niño, para esto se creó un módulo de notificaciones, que se activa cada vez que se detecta un riesgo, el módulo de notificaciones posee las siguientes características:

- Se hace el análisis cada 500 frames, esto se hace con la finalidad si cambia la situación y el entorno en que se presenta el riesgo para el niño.
- Si coindice que existe un niño y un objeto que represente un riesgo como un cuchillo, una herramienta o un contacto de luz, se toma el snapshot de esa imagen con los bounding box donde se identifica al niño y al objeto que representa un riesgo para el niño.
- Cuando se tiene el snapshot se envía como imagen adjunta al correo del padre o tutor.
- El proceso se repite de forma indefinida

Parte del código fuente para la detección de riesgos se observa en la Figura 35.

```
# Stream results
im0 = annotator.result()
if view_img:
    cv2.imshow(str(p), im0)
    if risk_detected and count > 500:
        LOGGER.info(f'------------- Sending email -------------')
        cv2.imwrite("./notifications/images/detection.jpg", im0)
        exec(open('./notifications/notification.py').read())
        count = 0
    cv2.waitKey(1)  # 1 millisecond
```

Figura 35 Fragmento de código para la detección de riesgos (Elaboración propia)

En la siguiente Figura 36 se observa un breve código para el envió de notificaciones por correo electrónico.

```
# Turn these into plain/html MIMEText objects
part = MIMEText(html, "html")

# Add HTML/plain-text parts to MIMEMultipart message
# The email client will try to render the last part first
message.attach(part)

# This example assumes the image is in the current directory
fp = open('./notifications/images/detection.jpg', 'rb')
msgImage = MIMEImage(fp.read())
fp.close()

# Define the image's ID as referenced above
msgImage.add_header('Content-ID', '<image1>')
message.attach(msgImage)

# Create secure connection with server and send email
context = ssl.create_default_context()
with smtplib.SMTP_SSL("smtp.gmail.com", 465, context=context) as server:
    server.login(sender_email, password)
    server.sendmail(
        sender_email, receiver_email, message.as_string()
    )
```

Figura 36 Código para él envió de notificaciones (Elaboración propia)

Él envió de notificaciones se puede observar en la Figura 37 como es que realmente se ve en la bandeja de correo electrónico del destinatario, para este caso el correo del padre o tutor

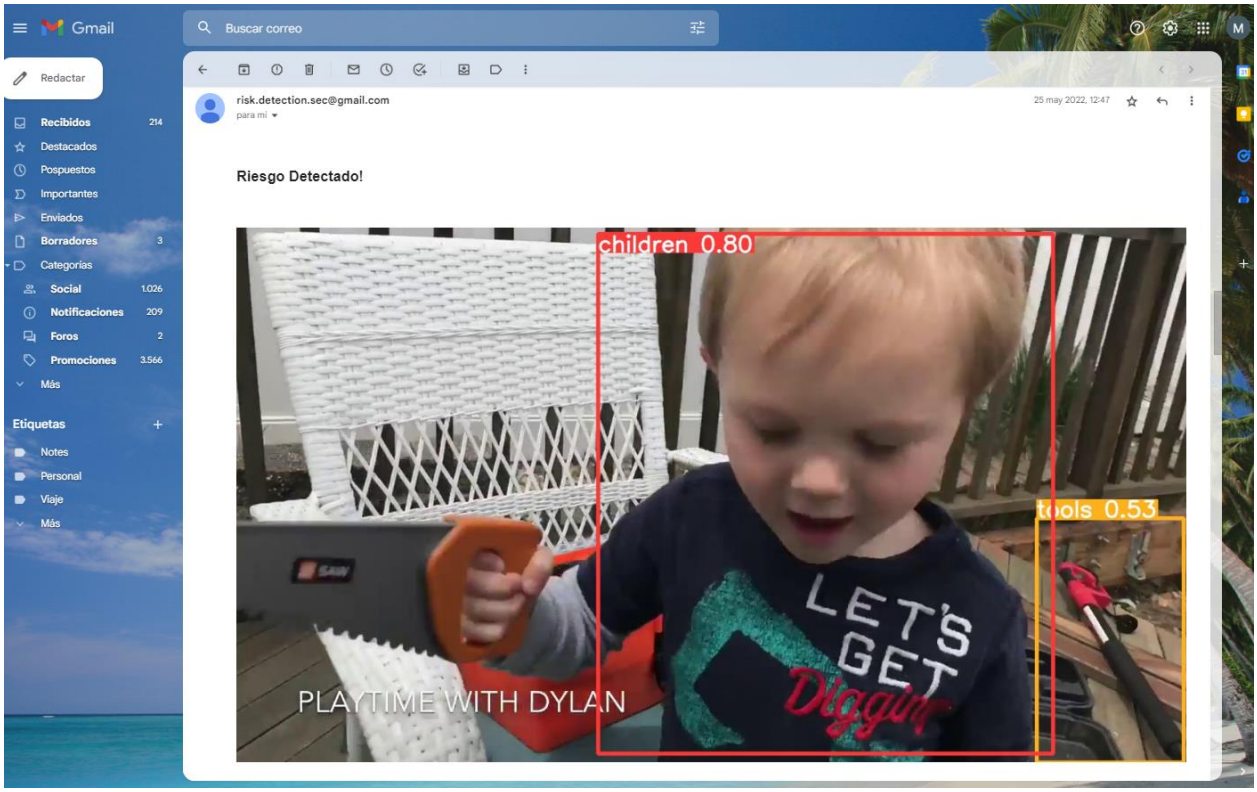

Figura 37 Envió de notificaciones (Elaboración propia)

De esta forma, es como por medio del sistema de detección de riesgos y la inteligencia artificial es como se pretende disminuir de forma drástica el número de accidentes que ocurren dentro del hogar, el monitoreo constante de las circunstancias, eventos y objetos que representan un riesgo, se puede prevenir un suceso lamentable antes que suceda.

Capítulo V. Resultados / Conclusiones

Resultados

Como resultado del entrenamiento se observa un comportamiento que va mejorando progresivamente en la precisión y el promedio de detección (mean Average Precision – mAP) al mismo tiempo las pérdidas o errores en el entrenamiento, van disminuyendo Figura 38.

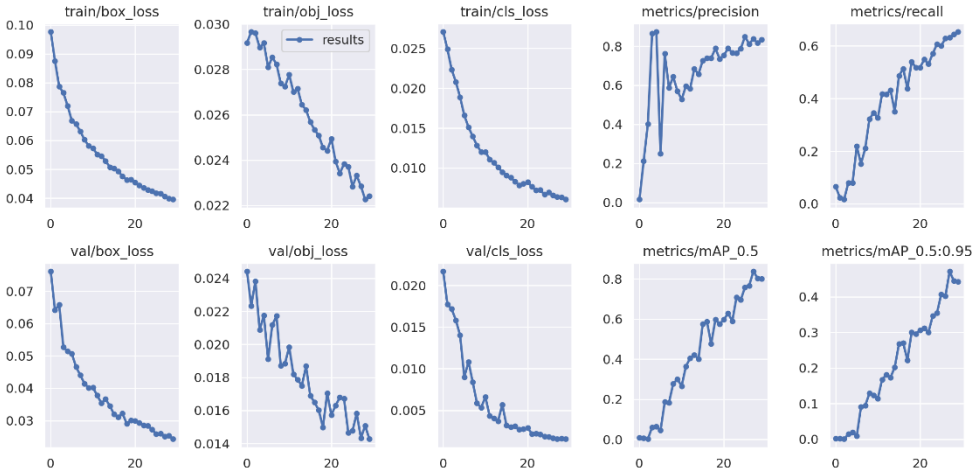

Figura 38 Resultados del entrenamiento (Elaboración propia)

Como ya se comentó en el capítulo anterior para comprobar los resultados se tomaron al azar 3 videos de la página de contenidas de YouTube (Youtube, 2022).

Dichos videos fueron empleados para comprobar los resultados y hacer las detecciones, los videos fueron analizados uno por uno, una vez detectado un riesgo, es decir que dentro de un frame del video se detecte al mismo tiempo un niño y un objeto peligroso, se crea un bounding box sobre el niño y el objeto peligroso, y los resultados de análisis del proceso se almacenan en la siguiente ruta:

```
yolov5/runs/detect/yolo_dangerous_detect[#]
```

El tipo de archivos almacenados dependerá de que tipo de archivo se este analizando, por ejemplo, si se analiza un video, se guardara un video, pero con un bounding box indicando las detecciones que fueron identificadas en el video, Figura 39 y Figura 40.

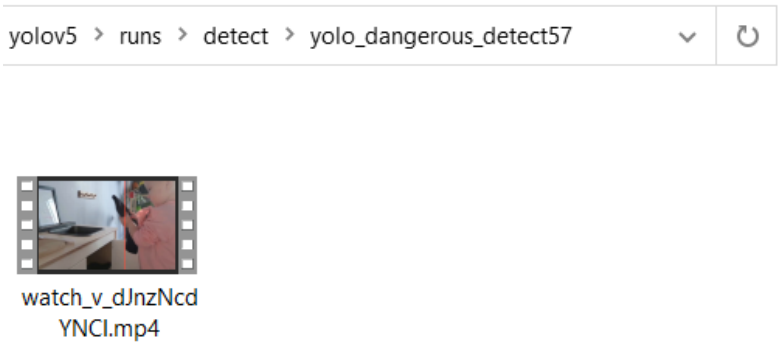

Figura 39 Archivo .mp4 generado con las detecciones (Elaboración propia)

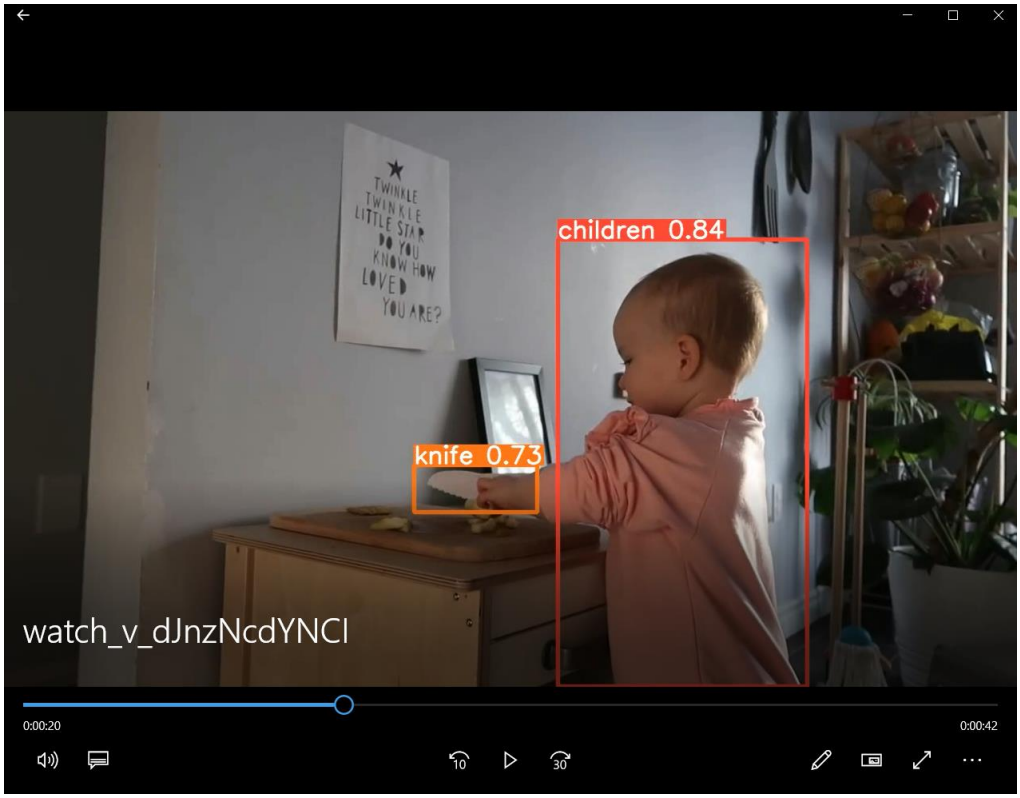

Figura 40 Reproducción del video con las detecciones (Elaboración propia)

Estos archivos se generan al final de todo el proceso de análisis del video, el tiempo de procesamiento dependerá de la duración del video, la resolución del video, y la capacidad del equipo de cómputo con el que se esté realizando el procesamiento. Como es de suponer entre mejor sea el equipo de cómputo, más rápido será el procesamiento de los videos o las imágenes.

La breve explicación que se dio en el capítulo anterior sobre el envió de notificaciones, durante el procesamiento de los videos para la detección de riesgos, en cada detección de riesgo se estuvo enviando un correo electrónico a la dirección de correo especificada en la configuración de la aplicación, que en teoría debe ser la dirección de correo de los padres o tutores del niño. Como observa en la Figura 41 la reproducción del video y la detección del riesgo

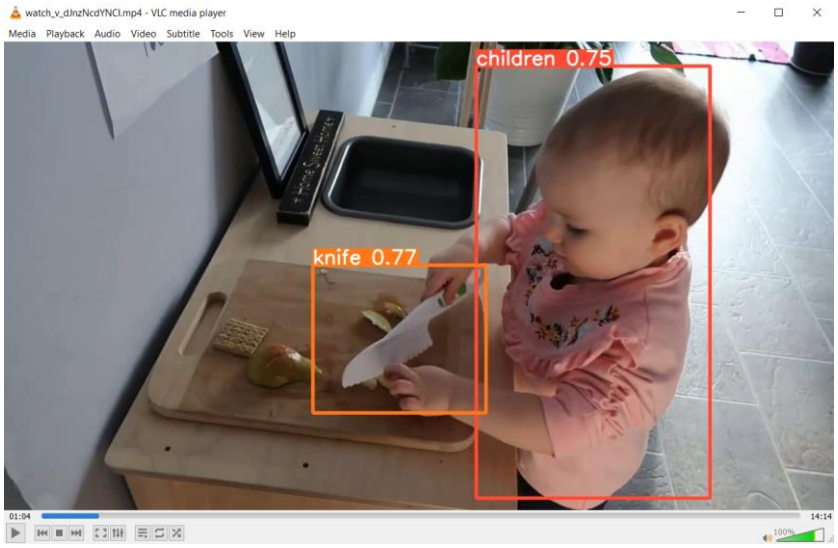

Figura 41 Reproducción del video (elaboración propia)

Cada correo se envía una imagen adjunta con el porcentaje de confidencialidad, de la detección Figura 42, 43, 44.

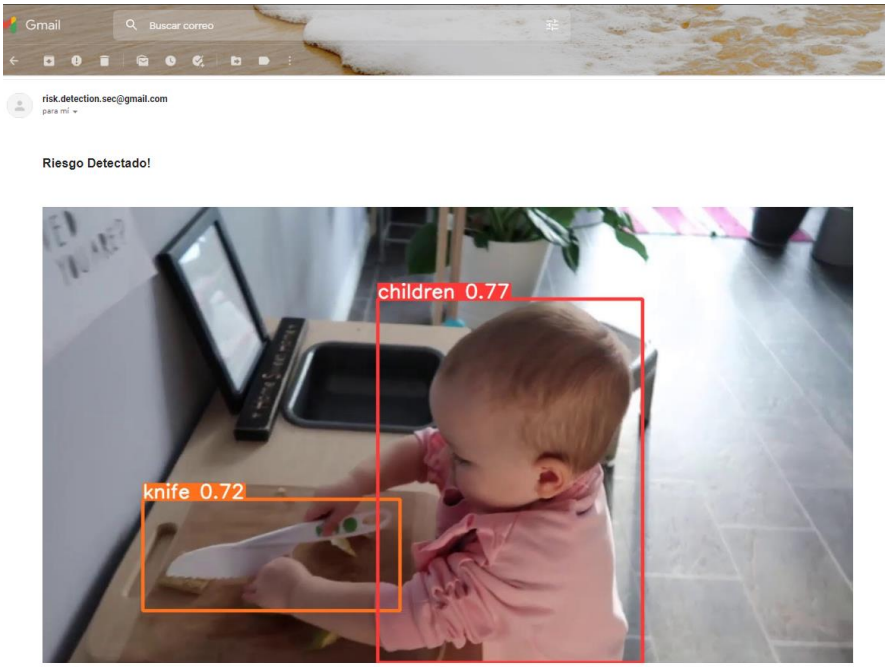

Figura 42 Notificación detección de niño con cuchillo (Elaboración propia)

Ejemplo para la detección de niños cercanos a un contacto de luz o de herramientas:

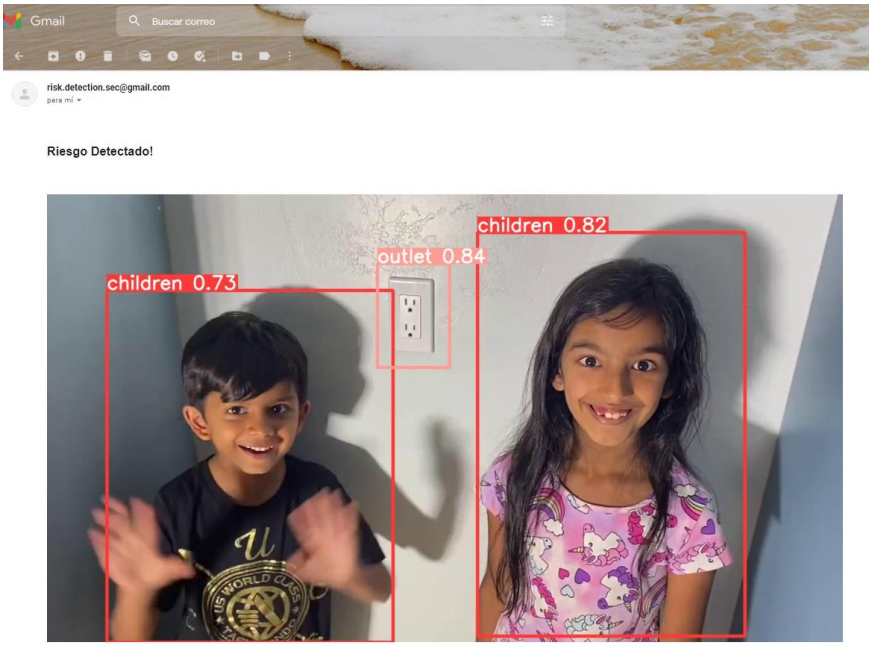

Figura 43 Notificación detección de niño con contacto de luz (Elaboración propia)

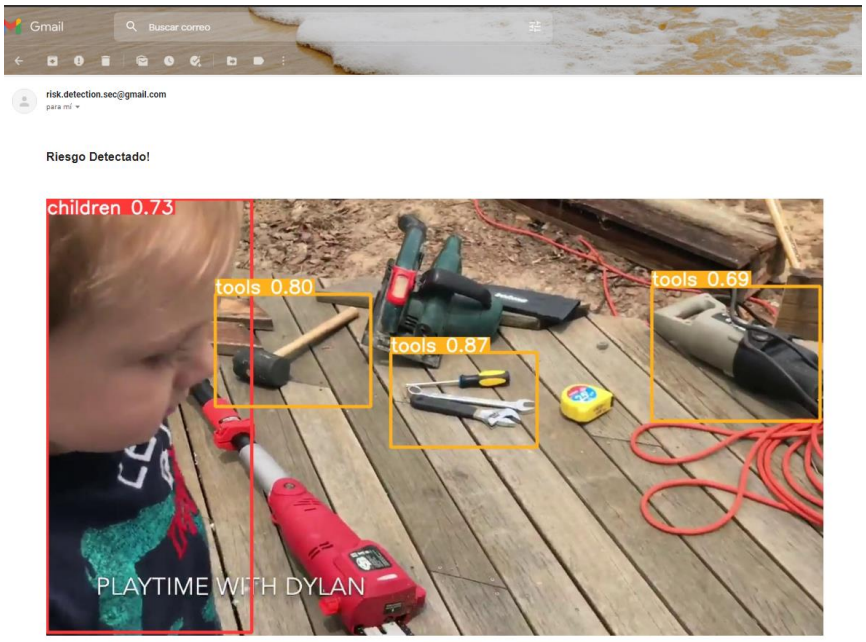

Figura 44 Notificación detección de niño con herramientas (Elaboración propia)

Los resultados fueron satisfactorios para esta prueba de concepto, con este sistema de inteligencia artificial se pretende reducir considerablemente el número de accidentes que ocurren dentro del hogar y salvaguardar la integridad física de los niños menores de 12 años.

Para obtener mejores resultados en la detección de personas es importante complementar el dataset con la mayor cantidad de imágenes posible de niños cuyas características sean del mismo tipo al que se desea

detectar, como es de saber, existe una extensa variedad de niños y niñas en el rango de 0 a 12 años, lo mismo aplica para los objetos, entre mayor sea la cantidad de imágenes de los objetos que se desea detectar, o delimitar solo unos cuantos objetos más comunes para una casa en particular, será mejor el entrenamiento y los resultados de las detecciones serán más satisfactorios.

Conclusiones

En la actualidad, ha avanzado enormemente el desarrollo de los algoritmos de machine learning y deep learning, para este proyecto de investigación se dio un enfoque a la visión por computadora, cabe destacar que una de las principales características que ha mejorado es el rendimiento y la precisión de los algoritmos que detectan objetos en las imágenes y videos, su optimización ha mejorado a tal grado que ahora es posible implementarlos de manera fácil y sencilla, en máquinas que no son especializadas para las áreas de computación gráfica.

Para nuestro propósito, crear un prototipo de detección de eventos que ponen en riesgo a los niños dentro del hogar, siguiendo los pasos mencionados anteriormente, resultó sencillo implementar todo un sistema capaz de hacer un análisis de videos e imágenes para detectar riesgos que se pueden sufrir dentro del hogar, sin embargo lo que resulto más complicado fue construir el dataset para entrenar el modelo de machine learning que será el encargado de hacer las detección de los objetos y las personas, esto debido a que las imágenes deben tener ciertas características en común, siempre y cuando esto sea posible, entre mayor sea la cantidad de imágenes y con este me refiero a mínimo 10 mil imágenes de cada clase con características en común, incrementara considerablemente la confidencialidad y la precisión en el entrenamiento y en las detecciones, en caso contrario si no entrenamos un algoritmo con los datos adecuados, los resultados serán pobres y la detecciones serán casi nulas y en otros casos dará lugar a falsos positivos, es decir detectara objetos que no serán necesariamente los esperados, por ello es importante construir un dataset confiable para que las detecciones de los eventos sea satisfactorio.

Otro de los aspectos que se investigaron con éxito fue la elección del framework de detección de objetos, con base al estado del arte, YOLO en la versión 5, resulto ser un framework altamente eficiente, debido a que no requiere de grandes recursos computaciones, es fácil de implementar y ofrece resultados sorprendentes. Recientemente fue lanzada la versión 7 y 8 de Yolo, a futuro sería bueno explorar las nuevas características que ofrecen estas versiones.

Este sistema tiene gran potencial y áreas de mejoras, una de ellas es la interconexión con dispositivos inteligentes que pueden ayudar a mejorar la seguridad de los niños, una vez detectada la eventualidad, de forma inmediata se puede activar mecanismos de seguridad como cerraduras, robots de limpieza, alarmas, etc., además de por supuesto de las notificaciones que se envían por correo electrónico a los padres o tutores.

Otra área de oportunidad interesante seria interconectar el sistema con asistentes de voz inteligentes, la idea que cuando se detecte un posible riesgo, se envié la notificación a Alexa para que el dispositivo este avisando por medio de voz a la familia y ellos estén enterados de que puede ocurrir algún accidente dentro del hogar, y mejor aún, implementar un mecanismo de face detection para que se pueda identificar sobre que niño en específico se encuentra en riesgo y así poder avisar por medio del asistente de voz y mencionar el nombre del niño que se encuentra en riesgo.

Todo esto es posible, pero requiere un mayor esfuerzo, tiempo y dedicación para hacer todo este entrenamiento en las redes neuronales, entre más sofisticado sea el sistema, también incrementará la seguridad de los niños.

Índice de Imágenes

Bibliografía

Alatorre, E. G. (02 de Marzo de 2021). *Las bases del desarrollo tecnológico en China*. Obtenido de El financiero: https://www.elfinanciero.com.mx/opinion/eugenio-gomez/las-bases-del-desarrollo-tecnologico-en-china/

Andalucia, F. (2010). *Accidentes en el hogar.* Obtenido de facua.org: https://www.facua.org/es/guia.php?Id=132

Anzóla, S. (2002). *Administración de Pequeñas Empresas de México.* México: MCGRAW-HILL / INTERAMERICANA DE MEXICO.

Bagnato , J. (29 de Noviembre de 2018). *¿Cómo funcionan las Convolutional Neural Networks? Visión por Ordenador*. Obtenido de https://www.aprendemachinelearning.com/: https://www.aprendemachinelearning.com/como-funcionan-las-convolutional-neural-networks-vision-por-ordenador/

Beilharz, P. (2001). Liquid modernity. *Contemporary Sociology, ProQuest, 30*(4), 420-421.

Berzal, F. (2018). *Redes Neuronales & Deep Learning.* Granada: Apache License.

Booch, G. (1996). *Análisis y diseño orientado a objetos* (Segunda Edición ed.). México: Addison Wesley Iberoamericana S.A.

Borjas. (24 de Septiembre de 2021). *¿Qué es una RNN? Red Neuronal Recurrente*. Obtenido de https://www.islabit.com/: https://www.islabit.com/152200/que-es-una-rnn-red-neuronal-recurrente.html

Calvo, D. (13 de Julio de 2017). *Clasificación de redes neuronales artificiales*. Obtenido de Diego Calvo: https://www.diegocalvo.es/clasificacion-de-redes-neuronales-artificiales/

Calvo, J. (20 de Julio de 2020). *Crear red neuronal desde las matemáticas*. Obtenido de European Valley: https://www.europeanvalley.es/noticias/crear-red-neuronal-desde-las-matematicas/

Cañadas, R. (22 de Noviembre de 2021). *Redes neuronales recurrentes*. Obtenido de https://abdatum.com/: https://abdatum.com/tecnologia/redes-neuronales-recurrentes

Chaos, I. (2021). *Redes Neuronales*. Obtenido de interactivechaos.com: https://interactivechaos.com/es/manual/tutorial-de-machine-learning/redes-neuronales

Clemens, W. (27 de Marzo de 2020). *How to recognise objects in videos with PyTorch*. Obtenido de Medium: https://medium.com/dida-machine-learning/how-to-recognise-objects-in-videos-with-pytorch-44f39f4c22f9

Clemens, W. (2020 de Marzo de 27). *Medium*. Obtenido de How to recognise objects in videos with PyTorch. : https://medium.com/dida-machine-learning/how-to-recognise-objects-in-videos-with-pytorch-44f39f4c22f9

Cochard, D. (29 de Marzo de 2021). *YOLOv5 : The Latest Model for Object Detection*. Obtenido de Medium: https://medium.com/axinc-ai/yolov5-the-latest-model-for-object-detection-b13320ec516b

Colon, A., & Moscaritolo, A. (26 de Abril de 2021). *The Best Smart Home Devices for 2021*. Obtenido de PC Magazine: https://www.pcmag.com/news/the-best-smart-home-devices

Dash, S., & Kumar, S. (2022). *Deep Learning Machine Learning and IoT in Biomedical and Health Informatics.* Florida: CRC Press.

eResearch. (26 de Noviembre de 2014). *Python Virtual Environments*. Obtenido de https://uoa-eresearch.github.io/: https://uoa-eresearch.github.io/eresearch-cookbook/recipe/2014/11/26/python-virtual-env/

Ferré Grau, X., & Sánchez Segura, M. I. (s.f.). Desarrollo Orientado a Objetos con UML. *Facultad de Informática – UPM*.

Foro, V. C. (2017). *Quora*. Obtenido de Quora: https://www.quora.com/What-is-the-best-algorithm-for-object-detection

Gonzalez Osorio, G. J. (10 de Febrero de 2017). Reconocimiento de objetos utilizando Open CV y Python en una Raspberry Pi 2 en una tlapalería. *Tesis de licenciatura, Universidad Autónoma del Estado de México*. Estado de México, Estado de México, México: Centro Universitario UAEM Texcoco.

Google. (3 de Junio de 2021). *Ooen Images Dataset*. Obtenido de Google Open Source: https://opensource.google/projects/open-images-dataset

Guzman, E. R. (2020). Detección de objetos con redes neuronales profundas para un robot de servicio. *Tesis de maestría, Universidad Nacional Autónoma de México*. México, México, México: Biorobotics Fi UNAM.

Haaf, W. (11 de Mayo de 2020). *What the Heck Is a Smart Home?* Obtenido de goodtimes.ca: https://goodtimes.ca/what-the-heck-is-a-smart-home/

IMSS. (30 de Diciembre de 2017). *IMSS Comunicación Social*. Obtenido de IMSS Prensa: http://www.imss.gob.mx/prensa/archivo/201712/398

INEGI, B. d. (n.i. de n.i. de 2014). *Esperanza de vida de los negocios.* Recuperado el 24 de 06 de 2016, de http://upla.zacatecas.gob.mx/wp-content/uploads/2014/06/BOLETINES/Esperanza%20de%20vida%20de%20los%20negocios.pdf

Jocher, G., & Koblanski, J. (2021). *Ultralytics*. Obtenido de GitHub: https://github.com/ultralytics/yolov5

Keshari, K. (25 de Noviembre de 2020). *Object Detection Tutorial in TensorFlow: Real-Time Object Detection*. Obtenido de Edureka: https://www.edureka.co/blog/tensorflow-object-detection-tutorial/

Khandelwal, R. (24 de Noviembre de 2019). *Yolov2 for Object detection from a video*. Obtenido de Towards Data Science: https://towardsdatascience.com/yolov2-for-object-detection-from-a-video-cd4574354d8e

Larman, C. (2004). *Applying UML and Patterns, An introduction to Object-Oriented Analysis and Design and the Unified Process* (Segunda Edición ed.). Prentice Hall.

Mateo Jiménez, M. (20 de Agosto de 2020). Reconocimiento facial como medida de seguridad para alertar el robo de automóviles. *Tesis de Licenciatura, Universidad Autónoma del Estado de México*. Estado de México, Estado de México, México: Centro Universitario UAEM Atlacomulco.

Meel, V. (17 de Marzo de 2021). *YOLOv5 Is Here! Is It Real or a Fake?* Obtenido de viso.ai: https://viso.ai/deep-learning/yolov5-controversy/

Montenegro, B., & Flores, M. (2022). Detección de peatones en el día y en la noche usando YOLO-v5. *Ingenius*, 11.

Mota Hernández, C. I., Contreras Troya, T. I., & Alvarado Corona, R. (2015). A systems methodology to solve economical-financial problems (SMEFP). *International Journal of Innovative Computing, Information and Control, 11*(1), 173-188.

Nelson, J., & Solawetz, J. (10 de Junio de 2020). *YOLOv5 is Here: State-of-the-Art Object Detection at 140 FPS*. Obtenido de Robo Flow: https://blog.roboflow.com/yolov5-is-here/

Neuro Hive. (06 de Noviembre de 2018). *New Datasets for 3D Object Recognition*. Obtenido de neurohive.io: https://neurohive.io/en/datasets/new-datasets-for-3d-object-recognition/

opengenus. (2022). *YOLO v5 model architecture*. Obtenido de iq.opengenus.org: https://iq.opengenus.org/yolov5/

Palma, J., & Marín, R. (2008). *Inteligencia Artificial Métodos, Técnicas y Aplicaciones.* España: McGraw-Hill.

Peden, M., Oyegbite, K., Ozanne-Smith, J., Hyder, A., Branche, C., Rahman, F., & Rivara , F. (2012). *Informe mundial sobre prevencion de las lesiones en los niños.* Washintong D.C. : Organizacion Panamericana de la Salud.

Pressman, R. (2010). *Ingeniería del Software. Un enfoque práctico, 7ª edición.* n.i.: Mc Graw Hill.

Redmon, Joseph, Farhadi, & Ali. (2018). *YOLO: Real-Time Object Detection*. Obtenido de Pjreddie: https://pjreddie.com/darknet/yolo/

Rouse, W. B. (2015). *Modeling and Visualization of Complex Systems and Enterprises: Explorations of Physical, Human, Economic, and Social Phenomena.* John Wiley & Sons.

Slim, F. C. (2009). *Accidentes en hogar, segunda causa de muerte infantil en México*. Obtenido de ClikiSalud: https://www.clikisalud.net/accidentes-en-hogar-segunda-causa-de-muerte-infantil-en-mexico/

Suaste , S., & Rosales , E. (1 de Noviembre de 2020). *Aplicación de redes neuronales convolucionales en el análisis de estructuras*. Obtenido de https://www.boletin.upiita.ipn.mx/: https://www.boletin.upiita.ipn.mx/index.php/ciencia/899-cyt-numero-81/1852-aplicacion-de-redes-neuronales-convolucionales-en-el-analisis-de-estructuras

Suaste Martínez, S. J., & Peña Alfaro, E. M. (2020). *Aplicación de redes neuronales convolucionales en el análisis de estructuras.* México: Instituto Politécnico Nacional.

Świeżewsk, J. (22 de Mayo de 2020). *YOLO Algorithm and YOLO Object Detection: An Introduction*. Obtenido de Appsilon: https://appsilon.com/object-detection-yolo-algorithm/

Ultralytics. (31 de Mayo de 2022). *https://github.com/ultralytics*. Obtenido de YoloV5: https://github.com/ultralytics/yolov5

Ultralytics. (2022). *Ultralytics/yolov5*. Obtenido de Github: https://github.com/ultralytics/yolov5

Velázquez, J. (2019). Identificación de peatones en imágenes aéreas con redes neuronales explicativas y fusión de sensores. *Tesis de maestría, Instituto Nacional de Astrofísica, Óptica y Electrónica*. Repositorio Institucional INAOE.

Velázquez, J. (2019). Identificación de peatones en imágenes aéreas con redes neuronales explicativas y fusión de sensores. *Tesis de maestría, Instituto Nacional de Astrofísica, Óptica y Electrónica*. Repositorio Institucional INAOE.

Von Bertalanffy, L. (1989). *Teoría General de los Sistemas* (Séptima reimpresión ed.). México: Fondo de Cultura Económica.
Youtube. (2022). *Youtube*. Obtenido de Youtube: https://www.youtube.com/