UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE INGENIERÍA EN SISTEMAS, ELECTRÓNICA E INDUSTRIAL

MAESTRÍA EN ELECTRÓNICA Y AUTOMATIZACIÓN, MENCIÓN CONTROL DE PROCESOS

Tema: Sistema electrónico de control y etiquetado de molduras para cuadros en tiempo-real mediante machine learning

---

Trabajo de Titulación previo a la obtención del Grado académico de Magíster en Electrónica y Automatización Mención Control de Procesos

Modalidad de Titulación ¨Proyecto de Desarrollo¨

Autor: Ing. David Alejandro Chávez Pico

Director: Ing. Marco Antonio Herrera Garzón, MSc.

Ambato – Ecuador

2022

APROBACIÓN DEL TRABAJO DE TITULACIÓN

A la Unidad Académica de Titulación de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial.

El Tribunal Receptor de la Defensa del Trabajo de Titulación, presidido por la Ingeniera Elsa Pilar Urrutia Urrutia Magíster, e integrado por los señores Ingeniero Carlos Diego Gordón Gallegos, Doctor e Ingeniero Mario Geovanny García Carrillo, Magister, designados por la Unidad Académica de Titulación de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial de la Universidad Técnica de Ambato, para receptar el Trabajo de Titulación con el tema: "SISTEMA ELECTRONICO DE CONTROL Y ETIQUETADO DE MOLDURAS PARA CUADROS EN TIEMPO-REAL MEDIANTE MACHINE LEARNING", elaborado y presentado por el señor Ingeniero David Alejandro Chávez Pico, para optar por el Grado Académico de Magister en Electrónica y Automatización Mención Control de Procesos; una vez escuchada la defensa oral del Trabajo de Titulación el Tribunal aprueba y remite el trabajo para uso y custodia en las bibliotecas de la Universidad Técnica de Ambato.

Firmado electrónicamente por:
ELSA PILAR URRUTIA

Ing. Elsa Pilar Urrutia Urrutia, Mg.

Presidente y Miembro del Tribunal de Defensa

Firmado electrónicamente por:
CARLOS DIEGO GORDON GALLEGOS

Ing. Carlos Diego Gordón Gallegos, PhD

Miembro del Tribunal de Defensa

Firmado electrónicamente por:
MARIO GEOVANNY GARCIA CARRILLO

Ing. Mario Geovanny García Carrillo, Mg.

Miembro del Tribunal de Defensa

AUTORÍA DEL TRABAJO DE TITULACIÓN

La responsabilidad de las opiniones, comentarios y críticas emitidas en el Trabajo Titulación presentado con el tema: “SISTEMA ELECTRÓNICO DE CONTROL Y ETIQUETADO DE MOLDURAS PARA CUADROS EN TIEMPO-REAL MEDIANTE MACHINE LEARNING”, le corresponde exclusivamente al: Ing. David Alejandro Chávez Pico, Autor bajo la Dirección del Ing. Marco Antonio Herrera Garzón, MSc., director del Trabajo de Titulación; y el patrimonio intelectual a la Universidad Técnica de Ambato.

Firmado electrónicamente por:
DAVID ALEJANDRO CHAVEZ PICO

Ing. David Alejandro Chávez Pico
AUTOR

Firmado electrónicamente por:
MARCO ANTONIO HERRERA GARZON

Ing. Marco Antonio Herrera Garzón, MSc.
DIRECTOR

DERECHOS DE AUTOR

Autorizo a la Universidad Técnica de Ambato, para que el Trabajo de Titulación, sirva como un documento disponible para su lectura, consulta y procesos de investigación, según las normas de la Institución.

Cedo los Derechos de mi Trabajo de Titulación, con fines de difusión pública, además apruebo la reproducción de este, dentro de las regulaciones de la Universidad Técnica de Ambato.

Firmado electrónicamente por:
DAVID
ALEJANDRO
CHAVEZ PICO

Ing. David Alejandro Chávez Pico
C.C. 1804375408

ÍNDICE GENERAL

Contenido

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

AGRADECIMIENTO

Agradezco, a Dios por guiar mi camino y por darme fortaleza a lo largo de todos estos años, a mis padres y mis hermanos por brindarme el apoyo incondicional en todo momento, a la Universidad Técnica de Ambato que aportó a mi formación profesional y personal, a mi Tutor Ing. Marco Antonio Herrera Garzón, MSc. por su valiosa colaboración para la culminación del presente Proyecto de titulación, finalmente gracias a los docentes Ing. Mario Geovanny García Carrillo, MSc y al Ing. Carlos Diego Gordón Gallegos, PhD. quienes contribuyeron con su asesoría durante el desarrollo del presente Proyecto.

Ing. David Alejandro Chávez Pico

DEDICATORIA

Dedico este Proyecto de Titulación a mis queridos padres Amparo Eugenia Pico Medina y Guilber Rodrigo Chávez Sánchez quienes me han entregado su apoyo, cariño y comprensión en todo momento a lo largo de mi vida en especial para el cumplimiento de esta meta que es muy importante tanto para mi como para mis queridos padres que siempre me han apoyado en cada instante de mi vida, que ha pesar de los diferentes retos que se han presentado y estoy muy agradecido, me han forjado para ser un hombre más feliz y fuerte.

Ing. David Alejandro Chávez Pico

UNIVERSIDAD TÉCNICA DE AMBATO
FACULTAD DE INGENIERÍA EN SISTEMAS, ELECTRÓNICA E INDUSTRIAL
MAESTRÍA EN ELECTRÓNICA Y AUTOMATIZACIÓN, MENCIÓN CONTROL DE PROCESOS

TEMA:

¨ SISTEMA ELECTRÓNICO DE CONTROL Y ETIQUETADO DE MOLDURAS PARA CUADROS EN TIEMPO-REAL MEDIANTE MACHINE LEARNING¨

AUTOR: Ing. David Alejandro Chávez Pico
DIRECTOR: Ing. Marco Antonio Herrera Garzón, MSc.

LÍNEA DE INVESTIGACIÓN

- Tecnologías de la información y sistemas de control.

FECHA: 07 de Abril del 2022

RESUMEN EJECUTIVO

En este trabajo se presenta el diseño de un sistema electrónico en tiempo real que realiza la detección de los diferentes tipos de molduras fabricadas a partir de análisis del procesamiento de imagen de sus diferentes superficies, siluetas y colores. La necesidad de utilizar este sistema electrónico de control y etiquetado de molduras para cuadros en tiempo-real mediante machine learning es para reducir los tiempos de producción e ir almacenando la cantidad de molduras fabricadas en una base de datos con el fin de evitar tiempos muertos por parte de los trabajadores y así aumentar la productividad en la fábrica. Además, se cuenta con las herramientas, software y hardware para poder realizarlo, en este caso se va a utilizar un dispositivo llamado Jetson Nano de NVIDIA para el procesamiento de imágenes con su respectiva cámara, el cual permite realizar aplicaciones de visión artificial. Por otra parte, este sistema de control permite etiquetar las molduras fabricadas con sus respectivas características mediante un código QR. Esto beneficiará la producción ya que la cantidad de moldura se incrementará, ya que se reducirán costos y existirá incremento en sus utilidades. Otro aspecto importante, es con este sistema se pretende obtener escalabilidad para un futuro debido a que existen diferentes sucursales donde se transporta las molduras y

ayudará a tener un control más exacto desde que sale hasta que llega la carga para evitar retrasos en el conteo como convencionalmente se lo ha hecho en los últimos años, estos datos se podrá llevar directamente al departamento de contabilidad quienes están a cargo de contabilizar la producción que se realice en la fábrica y la cantidad que se transporta a la matriz y cada una de las sucursales.

Descriptores: Sistema embebido, visión artificial, Jetson Nano, Python, NVIDIA, Sistema de control, machine learning, códigos QR, tiempo-real y etiquetado.

UNIVERSIDAD TÉCNICA DE AMBATO FACULTAD DE INGENIERÍA EN SISTEMAS, ELECTRÓNICA E INDUSTRIAL

MAESTRÍA EN ELECTRÓNICA Y AUTOMATIZACIÓN, MENCIÓN CONTROL DE PROCESOS

THEME:
"ELECTRONIC SYSTEM FOR CONTROL AND LABELING OF FRAMEWORK IN REAL-TIME THROUGH MACHINE LEARNING"

AUTHOR: Ing. David Alejandro Chávez Pico
DIRECTED BY: Ing. Marco Antonio Herrera Garzón, MSc.
LINE OF RESEARCH:

- Information technologies and control systems

DATE: April 07th, 2022

EXECUTIVE SUMMARY

In this work, a real-time system is presented that performs the detection of the different types of moldings made from the analysis of the image processing of their different surfaces, silhouettes and colors. The need to use this electronic system for the control and labeling of moldings for paintings in real time through machine learning is to reduce production times and store the number of moldings manufactured in a database in order to avoid downtime on the part of workers and thus increase productivity in the factory. In addition, it has the tools, software and hardware to be able to do it, in this case a device called NVIDIA's Jetson Nano will be used for image analysis with its respective camera, which allows artificial vision applications. On the other hand, this control system allows the moldings manufactured to be labeled with their specific characteristics by means of a QR code to make it functional and practical in the factory. This will benefit production since the amount of molding will increase, since costs will be reduced and there will be an increase in profits. Another important aspect is that with this system it is intended to have scalability for the future due to the fact that there are different branches where the moldings are transported and it will help to have a more exact control from the time the load leaves until the load arrives to avoid delays in counting as conventionally. It has been done in recent years, this data can be taken directly to the accounting department who are in charge of the amount of production

that is carried out in the factory and the amount that is transported to the headquarters and each of the branches.

Summary: Embedded system, artificial vision, Jetson nano, Python, NVIDIA, Control system, machine learning, QR codes, real-time and labeling

CAPÍTULO I

EL PROBLEMA DE INVESTIGACIÓN

1.1 Introducción

El sistema electrónico de control y etiquetado de molduras para cuadros en tiempo real mediante machine learning ayudará a mejorar los tiempos de producción, lo cual beneficiará al empleador y empleados con una mejor logística debido a que se automatizará el último proceso antes de transportar el producto final a las diferentes sucursales, para lo cual se utilizará tecnología de sistemas embebidos. Este trabajo está dividido en cinco capítulos los cuales se detallan a continuación:

El Capítulo I se enfoca en la justificación donde se explicará la importancia, la originalidad del trabajo, el impacto y los beneficiarios del proyecto, de igual manera se presentan los objetivos cuales delimitarán el trabajo realizado.

En el Capítulo II, se tratará sobre el estado del arte o base científica que sustente el proyecto que se desea implementar, aquí se puede constatar que tipo de investigación intervendrán y así ayudar a entender el problema de investigación.

En el Capítulo III, se explicará sobre el tipo de metodología que se va a usar en el proyecto de titulación los cuales puntualizarán con exactitud el lugar donde se va a realizar el trabajo, que equipos y materiales se van a utilizar, cual es la hipótesis, procesamiento y análisis estadístico y como parte final los resultados alcanzados.

El Capítulo IV, se presenta el hardware y software que se va utilizar para el diseño e implementación del sistema electrónico para ser presentado de manera clara, precisa, integrada y secuencial con sus respectivos gráficos, esquemas, fotos, los cuales serán auto explicativos.

El Capítulo V presenta las conclusiones y recomendaciones de la investigación bibliográfica, siendo esta la base fundamental para el mejor entendimiento al problema planteado. Las recomendaciones indican las limitaciones obtenidas del prototipo durante el proceso de investigación y pruebas, logrando así dar más soporte a los objetivos trazados al inicio.

1.2 Justificación

El sistema electrónico de control que se desea realizar es práctico, útil e importante porque la fábrica de molduras requiere mejorar los tiempos de producción, control de calidad y manufactura. Es muy conveniente su implementación debido al factor tiempo y dinero, ya que los trabajadores sean han mantenido por un largo período de tiempo midiendo, etiquetando y contando de forma manual; esto conlleva mucho retardo de tiempo al momento de producir en grandes cantidades de manera más organizada y controlada. Con el presente trabajo, se espera que influya de manera positiva la automatización en control de procesos para el sector artesanal con el reconocimiento de patrones de cada modelo de moldura en tiempo real mediante machine learning, esto motivará a mejorar otras etapas que van desde el inicio que sería la elaboración de los modelos de moldura hasta el proceso final, lo cual sería la venta al público.

A nivel nacional esto ayudará a desarrollar proyectos similares con el fin de mejorar los procesos de control en las fábricas que no solamente se elaboren tiras de molduras sino pueden ser otro tipo de productos, esto tendrá un gran impacto para el país ya que esto ayudará agregando valor a la fabricación de productos mediante innovación para automatizar muchos procesos. Muy importante que en la práctica beneficia a muchos empleadores y empleados ya que ayudará a dinamizar la economía del país y la región, habrá más mano de obra. El presente trabajo se relaciona con el estudio de la maestría porque pertenece a la automatización relacionado al control de procesos para mejorar los tiempos de producción. La finalidad que persigue en este tipo de sistema es que pueda ser implementado en procesos que requieran su utilización, procesos similares, sobre todo al momento de detallar un producto, este pueda ser reducido en su máxima capacidad para el ahorro de tiempo y dinero. Los resultados se verán mejorados a través de la recolección de datos con el sistema inicial y compararlo con el nuevo sistema implementado, para verificar que el sistema actual sea efectivo. Además, esto permitirá mejorar las estrategias de almacenamiento, conteo de stock, transporte y manufactura.

1.3 Objetivos

1.3.1 General

- Diseñar e Implementar un sistema electrónico de control y etiquetado de molduras para cuadros en tiempo real mediante machine learning.

1.3.2 Específicos

- Revisar el estado del arte de machine learning relacionada a procesos similares al propuesto.
- Analizar y definir los algoritmos relacionados a machine learning y visión artificial para el reconocimiento de patrones.
- Implementar el sistema electrónico de control y etiquetado de molduras para cuadros en tiempo real mediante machine learning.

CAPÍTULO II

ANTECEDENTES INVESTIGATIVOS

2.1 Aprendizaje profundo

El aprendizaje profundo pertenece al conjunto de la inteligencia artificial y el aprendizaje automático, estas utilizan redes neuronales artificiales de muchas capas, lo cual brinda una gran precisión en tareas como clasificación de imágenes, reconocimiento de objetos, detección de objetos, reconocimiento de voz, traductor de idiomas, entre otras.[1]

El aprendizaje profundo tiene la capacidad de aprender automáticamente representaciones a partir de datos como imágenes, videos, texto o conocimientos de dominio humano. Mientras más datos se le proporcione al sistema, sus arquitecturas altamente flexibles pueden aprender y aumentar su precisión predictiva.[1]

Además, sus aplicaciones pueden ser en visión artificial, inteligencia artificial conversacional y sistemas de recomendación. Por lo cual existen aplicaciones de visión por computadora para el aprendizaje sea mediante conocimientos por imágenes y videos digitales. Otra sería aplicaciones de inteligencia artificial conversacional, estas en cambio ayudan a las computadoras comprender y comunicarse mediante lenguaje natural como se observa en la Fig.1.[1]

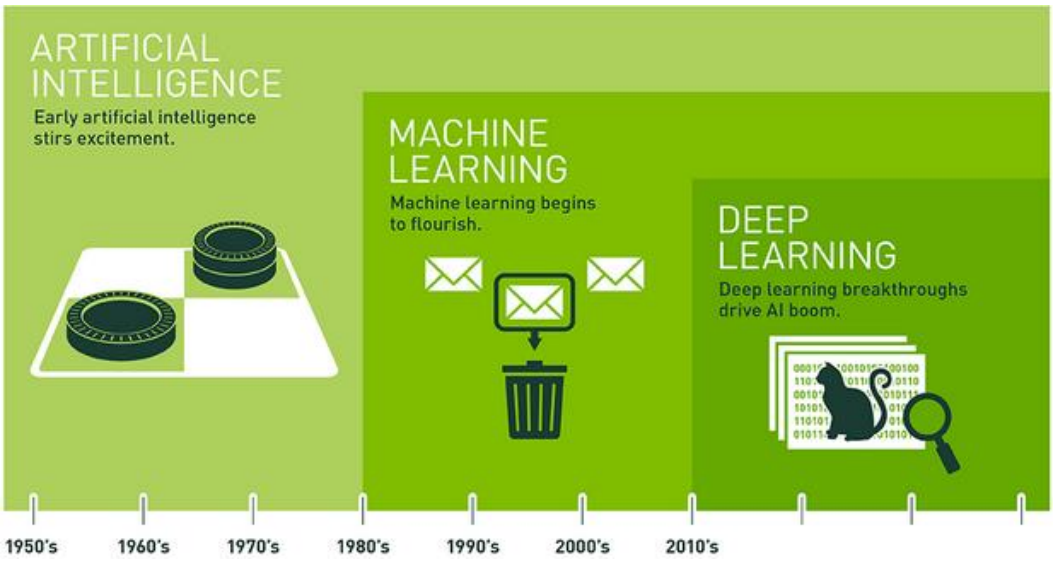

Fig. 1 Inteligencia Artificial [1]

Por otro lado, el aprendizaje profundo ha llevado a un mundo de grandes posibilidades dentro de la inteligencia artificial como AlphaGo, este es un programa desarrollado po Google DeepMind para jugar el juego de mesa Go, el cual se convirtió en la primera máquina de Go en ganar a un jugador profesional en Octubre del 2015, otra aplicación es asistentes de voz inteligentes, autos sin conductor y mucho más. Debido a estos sistemas que van avanzando, se han creado los GPU (Unidad de procesamiento gráfico) en la empresa mutinacional NVIDIA para aprendizaje profundo ya que estas unidades de procesamiento gráfico pueden acelerar significativamente el entrenamiento de aprendizaje profundo, lo cual ahorra tiempo porque tomaría días y semanas entrenar el sistema y con estas unidades llevaría solamente a horas y días. Si ya se tiene los modelos listos para su implementación, los desarrolladores pueden confiar las diferentes plataformas de inferencia acelerada por GPU para la nube, esto lleva a que exista un alto rendimiento y baja latencia para las redes neuronales profundas intensivas en computación.[1]

2.1.1 Redes Neuronales

Dentro de la revisión del estado del arte, uno de los campos muy estudiados es la Inteligencia Artificial, la cual corresponde al estudio de redes neuronales (RNAs), a estas se las entiende como redes en las que existen elementos procesadores de información dependiendo del comportamiento del sistema.[2]

Las RNAs tratan de comprender el comportamiento del cerebro del ser humano, tomando en cuenta el aprendizaje mediante la experiencia y adquisición de conocimiento genérico a través de un conjunto de datos obtenidos. Este tipo de sistemas imitan esquemáticamente la forma o estructura neuronal del cerebro humano, puede ser mediante un programa de ordenador (simulación), o también a través de un modelamiento de estructuras de procesamiento en paralelo (emulación) o también de forma física, cuyos sistemas tengan una arquitectura similar a la red neuronal biológica (implementación).[2]

Von Neumann es el tipo de arquitectura que tienen las computadoras, esta arquitectura está basada en microprocesadores muy rápidos capaces de ejecutar en serie instrucciones complejas de manera fiable, en cambio el cerebro está compuesto por

millones de procesadores elementales o neuronas, interconectadas entre sí formando redes. Esto lleva a que las neuronas no necesitan ser programadas, sino que aprenden a partir de los estímulos que reciben del entorno y operan masivamente en un esquema en parallel, como se observa en la Fig. 2.[2]

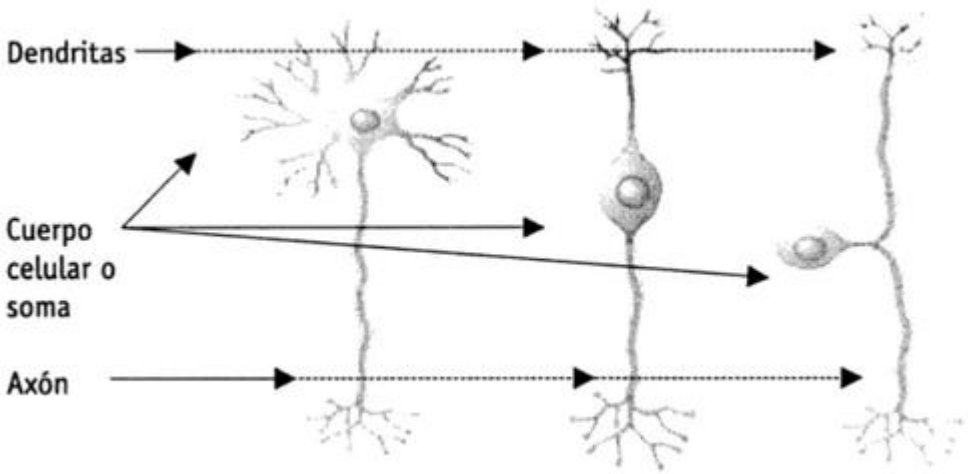

Fig. 2 Estructuras típicas de las neuronas biológicas [2]

2.1.2 Elementos y características principales de una red neuronal artificiales

Las RNAs tienen una estructura formada por una serie de procesadores elementales que forman dispositivos simples de cálculo que tienen un vector de entrada que viene del mundo exterior, con estos estímulos que son recibidos de otras neuronas, proporcionan una respuesta única (salida). Por lo cual existen tres tipos de neuronas artificiales: neuronas de entrada, neuronas ocultas y neuronas de salida, como podemos apreciar en la Fig.3 [2]

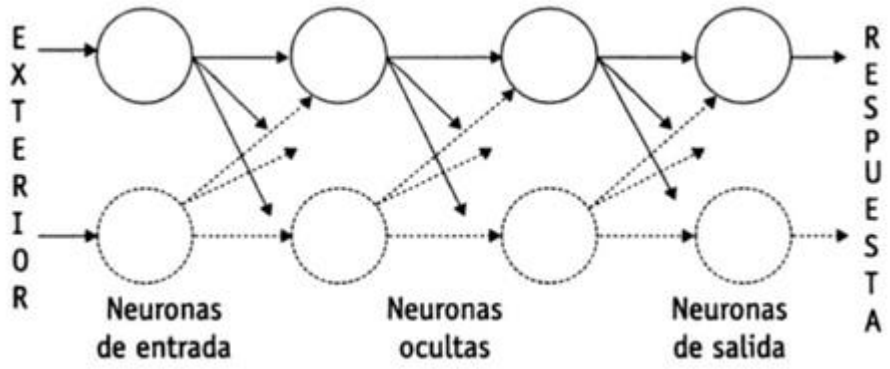

Fig. 3 Tipos de neuronas artificiales [3]

2.1.3 Funciones de activación de la neuronal

La función de activación o transferencia tiene carácter determinista, esto hace que la mayor parte de los modelos es monótona creciente y continua respecto al nivel de excitación de la neurona. Podemos encontrar seis funciones de transferencia: [3]

2.1.3.1 Función lineal o identidad

Esta función de transferencia devuelve directamente el valor de activación de la neurona. Es muy utilizada en redes de baja complejidad como el modelo Adaline, como se observa en la Fig. 4.[3]

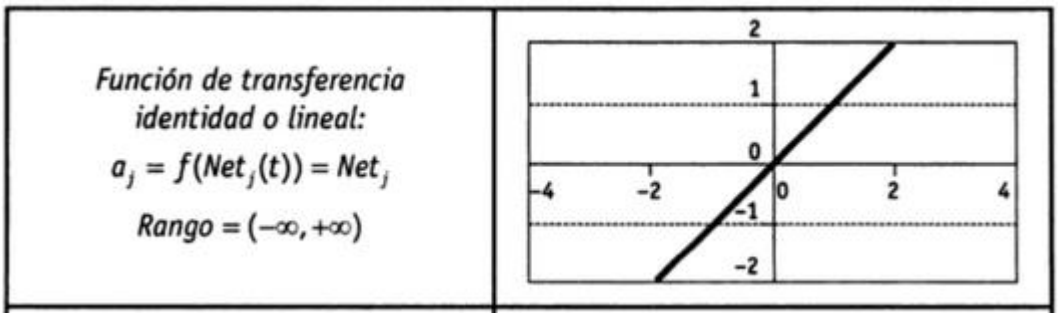

Fig. 4 Función de transferencia lineal [3]

2.1.3.2 Función escalón o signo

Este tipo de función presenta salidas binarias (normalmente {0,1} o {-1,1}). Por lo cual si la activación de la neurona es menor al valor de umbral, la salida se asocia a un determinado objetivo y si la salida es igual o mayor al umbral, esta se asocia a otro objetivo como lo veremos en la Fig. 5. En este tipo de función de activación en distintas aplicaciones son limitadas por lo que se trata de problemas binarios, una de las redes que se ha implementado y resulta útil es el perceptrón simple.[3]

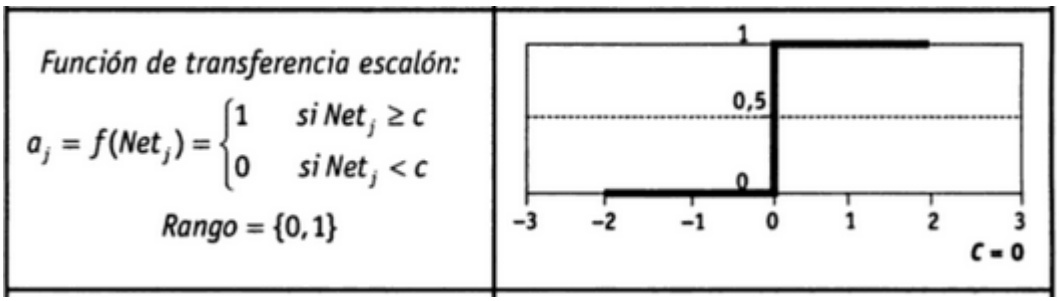

Fig. 5 Función escalón o signo [3]

2.1.3.3 Función mixta o lineal a tramos

Esta función de activación es una variante progresiva de la función escalón debido a que, si la activación de la unidad es menor que un límite inferior preestablecido, esta salida será determinado de un valor, pero si la activación es igual o mayor, la salida se determina por otro valor como lo vamos a ver en la Fig. 6. Cuando se toma como alternativa esta función puede considerarse como función lineal saturada en sus extremos, esto resulta de menos complejidad computacional y resulta más plausible desde el punto biológico.[3]

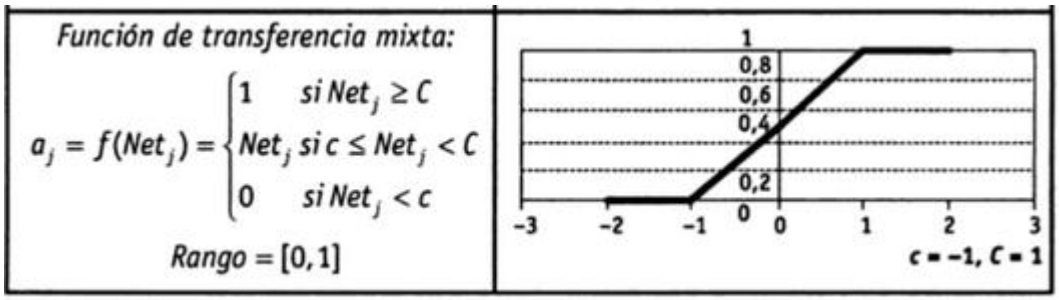

Fig. 6 Función de transferencia mixta [3]

2.1.3.4 Función sigmoidea

Esta función de activación muestra una progresión temporal con límites superiores e inferiores y esto lleva a las funciones de transferencia sigmoidea más utilizadas como función sigmoide o logística, la función tangente hiperbólica y la función sigmoidea modificada propuesta por AZOFF (1994). Estas funciones sigmoideas se caracterizan por el uso de una derivada siempre positiva y de igual manera a cero en sus límites asintóticos. Este tipo de funciones admiten aplicaciones para el aprendizaje similar al escalón, pero permiten emplear algoritmos de entrenamiento más avanzados como se puede observar en la Fig. 7.[3]

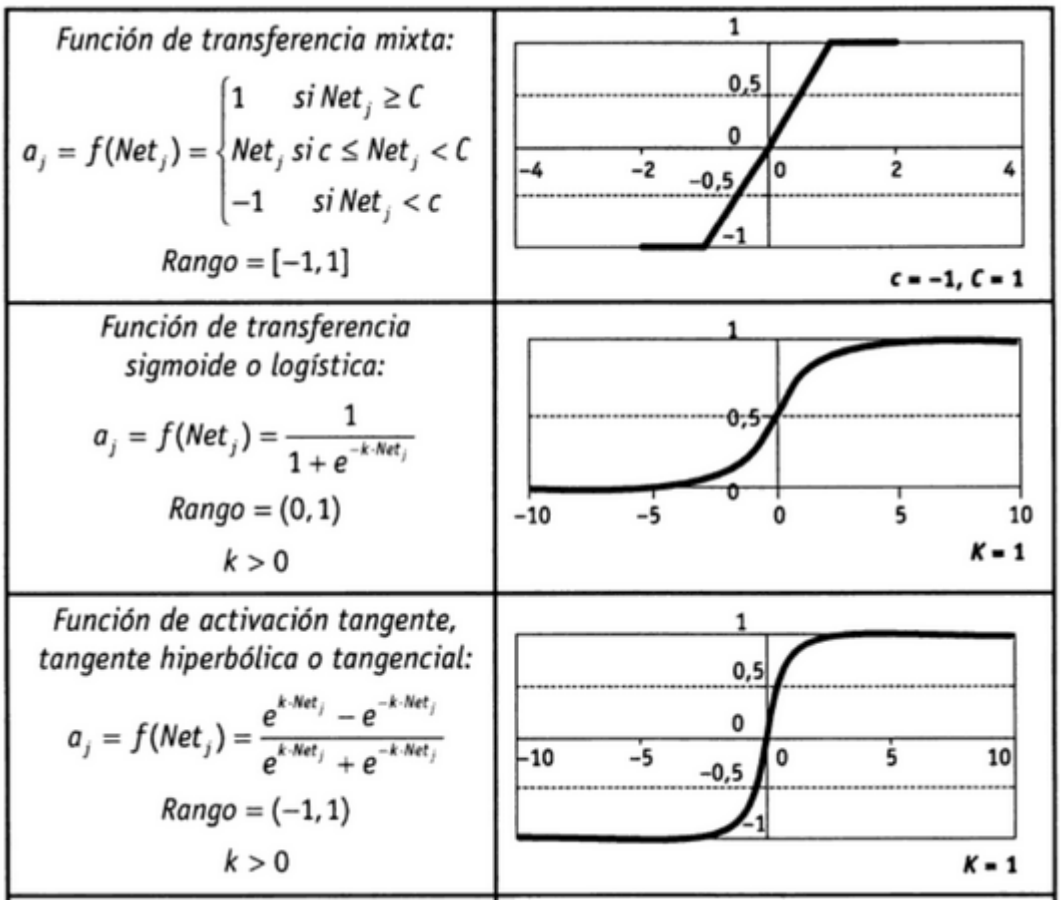

Fig. 7 Funciones de activación sigmoideas [3]

2.1.3.5 Función gaussiana

Este tipo de función de activación neuronal se adapta a la forma de una campana de Gauss cuyo radio, centro y apuntamiento son susceptibles de adaptación, lo que resulta muy versátiles. Este tipo de función son muy empleadas en redes complejas con m capas ocultas (m≥2), esto es utilizado en reglas de propagación que se basan en el cálculo de distancias cuadráticas entre los vectores de entrada y los pesos de la red. Esto se lo puede abservar en la Fig. 8 [3]

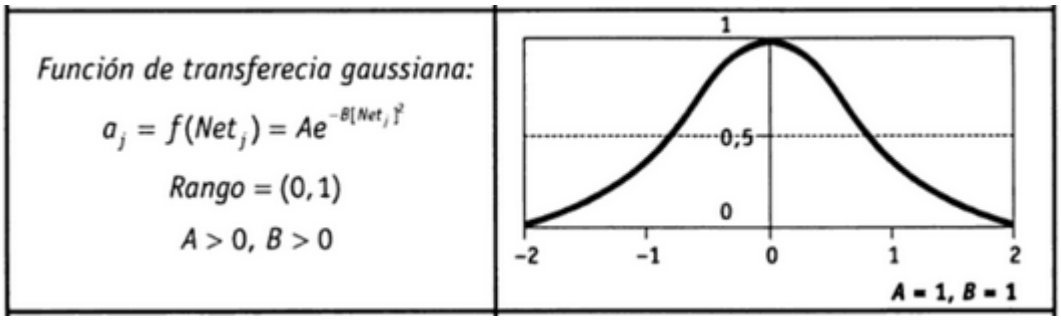

Fig. 8 Función de transferencia gaussiana [3]

2.1.3.6 Función de transferencia sinusoidal

Este tipo de función de activación genera salidas continuas en el intervalo [-1, +1] como en la Fig. 9. Son muy útiles al emplearse en los casos que requieren explícitamente una periodicidad temporal. [3]

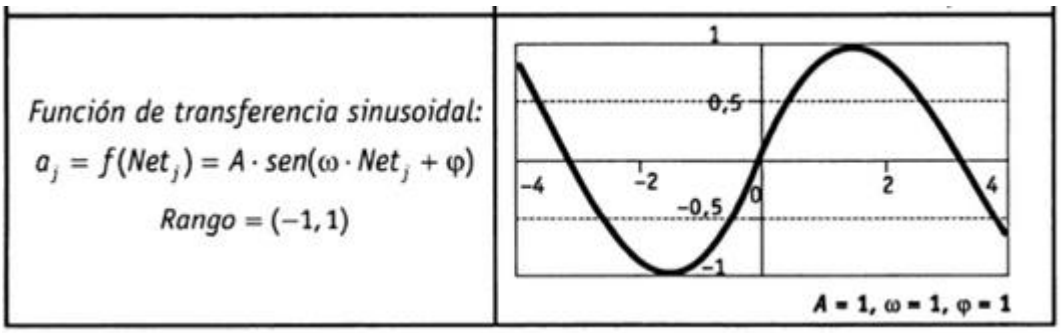

Fig. 9 Función de transferencia sinusoidal [3]

2.1.4 Detección de objetos

Mucho de los detectores de objetos basados en redes neuronales convolucionales (CNN) son aplicables en diferentes sistemas de recomendación. Por ejemplo, se tiene búsqueda de estacionamientosa través de cámaras de video urbanas, estos son ejecutados por modelos lentos y precisos, mientras para la colisión de automóviles, la advertencia viene dada por modelos inexactos y rápidos. El tiempo-real es una variable muy importante en sistemas de recomendación que generen pistas, sino también para la gestión de procesos autónomos y la reducción de la participación humana. [4]

La velocidad paradetectar objetos dentro de un sistema operativo es el principal objetivo dentro de lo que son sistemas de producción y optimización para cálculos computacionales en paralelo. Se ha realizado pruebas de funcionamiento y se han obtenido resultados de YOLOv4 las cuales se muestran en la Fig. 10. [4]

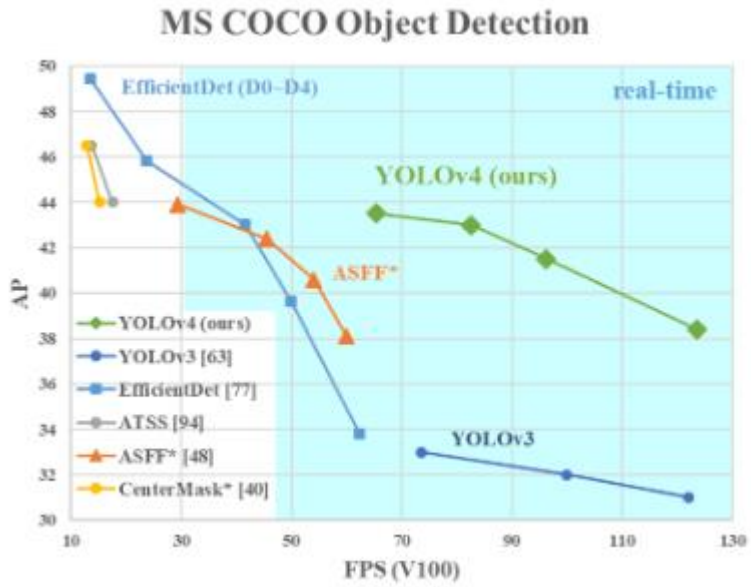

Fig. 10 Comparación entre YOLOv4 y otras propuestas de detectores de objetos [4]

A continuación, se han obtenido las siguientes contribuciones:

1. Se ha desarrollado un detector de objetos más eficaz, el cual hace más potente y poder usar como es la 1080 Ti la cual es usada en tarjetas de videojuegos por lo que utilizan unidadesde procesamiento gráfico más potentes, esto lo fbrica la empresa multinacional NVIDIA.

2. Se ha verificado el estado del arte entre Bag-of-Freebies y Bag-of-Specials, estos son entrenamientos heurísticos de detección de objetos los cuales tienen más complejidad en redes neuronales, pero mejoran en precición en un 5% en relación a otros modelos clasificación de objetos como puede ser Faster R-CNN y YOLOv3. [4]

2.1.4.1 Modelos de detección de objetos

Los modelos de detección de objetos están formados por dos partes, la primera es el backbone (columna vertebral) en el cual es el pre-entrenamiento en ImageNet y la cabecera la cual utiliza clases predictivas para los cuadros limitadores de objetos. [4]

Dentro del backbone que se ejecutan en la plataforma GPU se puede encontrar VGG, ResNet, ResNeXt o Densenet, encambio para detectores que se ejecutan en CPU se pueden encontrar SqueezeNet, mobileNet o ShuffleNet. En cambio, los que se refieren a la cabecera (head) se tiene dos tipos de detectores, el uno presenta dos etapas, a estas se pueden incluir R-CNN rápido y R-CNN más rápido, R-FCN y Libra R-CNN. También posible hacer dos detectores de objetos de escenario sin anclajes como es el RepPoints. Y el otro sería el de una etapa como es el YOLO, SSD y RetinaNet. Y de

una etapa sin ancla seriía de tipo CenterNet, CornerNet,FCOS. Actulmente existen otros tipos de aliados para recopilar información entre backbone y head, esto ayuda a obtener las características de las diferentes etapas, las redes equipadas con este mecanismo incluyen Feature Pyramid Network (FPN), Path Aggregation network (PAN), BiFPN y NAS-FPN. Y muchos de los investigadores utilizan un backbone DetNet, DetNAS y un nuevo modelo completo (SpineNet, HitDetector para la detección de objetos. Se puede visualizar la precición en relación a los distintos modelos de aprendizaje en la Fig. 11 [4]

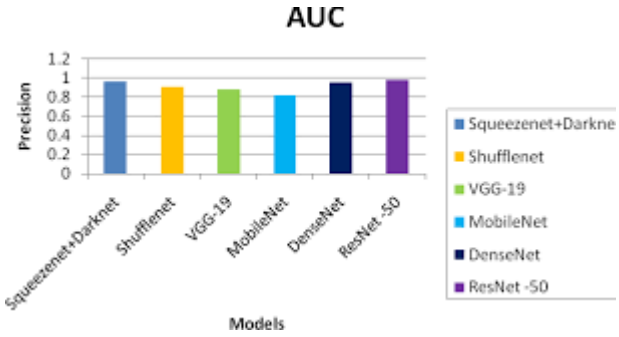

Fig. 11: Modelos de detección de objetos [4]

2.1.5 Gestor de trabajo (ANACONDA)

Este gestor de trabajo facilita la escritura de complejos algoritmos paralelos para la ejecución de tareas por lo cual se ha creado esta plataforma para más de 25 millones de usuarios en todo el mundo, la Edición Individual (Distribución) de código abierto es la forma más fácil de realizar ciencia de datos Python / R y aprendizaje automático en una sola máquina. Es desarrollado para practicantes en solitario, debido a que tiene un conjunto de herramientas que lo equipa para trabajar con miles de paquetes y bibliotecas de código abierto. [5]

2.1.5.1 Código abierto

Anaconda Individual Edition es la plataforma de distribución de Python más popular del mundo con más de 25 millones de usuarios en todo el mundo. Existen muchos paquetes de código abierto, es la plataforma elegida para la ciencia de datos de Python. [5]

2.1.5.2 Paquetes Conda

Tiene su repositorio en la nube del servidor para encontrar e instalar más de 7500 paquetes de ciencia de datos y aprendizaje automático. Con el comando conda-install,

puede comenzar a usar miles de paquetes de código abierto Conda, R, Python y muchos otros. [5]

2.1.5.3 Gestionar entornos

Individual Edition es una solución flexible de código abierto que proporciona las utilidades para crear, distribuir, instalar, actualizar y administrar software de manera multiplataforma. Conda facilita la gestión de múltiples entornos de datos que se pueden mantener y ejecutar por separado sin interferencias entre sí. [5]

2.1.5.4 Crear modelos de aprendizaje automático

Cree y entrene modelos de aprendizaje automático con los mejores paquetes de Python creados por la comunidad de código abierto, incluidos scikit-learn, TensorFlow y PyTorch.[5]

2.1.6 Entorno de deep learning (Jupyter Notebook)

JupyterLab es un entorno de desarrollo interactivo basado en web para cuadernos, código y datos. Su interfaz flexible permite a los usuarios configurar y organizar flujos de trabajo en ciencia de datos, informática científica, periodismo computacional y aprendizaje automático. Un diseño modular permite extensiones que amplían y enriquecen la funcionalidad.

Jupyter Notebook es una aplicación web para crear y compartir documentos que contienen código, visualizaciones y texto. Se puede utilizar para ciencia de datos, modelado estadístico, aprendizaje automático y mucho más.

Existen aplicaciones para las siguientes actividades

- Los cuadernos se pueden compartir con otras personas mediante el correo electrónico, Dropbox, GitHub y el visor de cuadernos de Jupyter.
- Su código puede producir una salida rica e interactiva: HTML, imágenes, videos, LaTeX y tipos MIME personalizados.
- Tiene herramientas de big data, como Apache Spark, de Python, R y Scala. Existen librerias para pandas, scikit-learn, ggplot2, TensorFlow.
- Es una herramienta utilizada para laboratorios, aulas virtuales
- Utiliza Docker y Kubernetes para escalar su implementación, aislar los procesos de los usuarios y simplificar la instalación del software debido a que trabajan como máquinas virtuales.

- Implementa Notebook junto a sus datos para brindar administración de software unificada y acceso a datos dentro de su organización.[6]

2.1.7 Entorno con GPUs (Google Colab)

Colaboratory, también llamado "Colab", permite ejecutar y programar en Python en el navegador con las siguientes ventajas:

- No requiere configuración
- Da acceso gratuito a GPUs
- Permite compartir contenido fácilmente

Los cuadernos de Colab te permiten combinar código ejecutable y texto enriquecido en un mismo documento, además de imágenes, HTML, LaTeX y mucho más. Los cuadernos que creas en Colab se almacenan en tu cuenta de Google Drive. Se pueden compartir cuadernos de Colab fácilmente, lo que permite comentarlos o incluso editarlos. [7]

2.1.7.1 Ciencia de datos

El análisis de grandes cantidades de información con la ayuda de la inteligencia artificial, se puede aprovechar toda la potencia de las bibliotecas de Python para analizar y visualizar datos. La celda de código de abajo utiliza **NumPy** para generar datos aleatorios y **Matplotlib** para visualizarlos. Para editar el código, solo se tiene que hacer clic en la celda.

Se puede importar datos a los cuadernos de Colab desde una cuenta de Google Drive, incluidas las hojas de cálculo, y también desde GitHub y muchas fuentes más. Para obtener más información sobre cómo importar datos y cómo se puede usar Colab en la ciencia de datos. [7]

2.1.7.2 Aprendizaje automático

Con Colab, se puede importar un conjunto de datos de imágenes, entrenar un clasificador de imágenes con dicho conjunto de datos y evaluar el modelo con tan solo usar unas pocas líneas de código. Los cuadernos de Colab ejecutan código en los servidores en la nube de Google, permite aprovechar la potencia del hardware de Google, incluidas las GPU y TPU, independientemente de la potencia del equipo de cómputo. Lo único que se necesita es un navegador.

Colab es una herramienta muy utilizada en la comunidad de aprendizaje automático. Estos son algunos ejemplos de las aplicaciones que tiene Colab:

- Dar los primeros pasos con TensorFlow
- Desarrollar y entrenar redes neuronales
- Experimentar con TPUs
- Divulgar datos de investigación sobre IA
- Crear tutoriales [7]

2.1.8 Hardware para redes neuronales

Para el entrenamiento de redes neuronales convolucionales en la detección de objetos, se va utilizar modelos de entrenamiento preentrenados por parte de la empresa multinacional NVIDIA, para lo cual se va necesitar analizar las características de las diferentes tarjetas que tienen GPU y CPU, por lo cual se va mostrar algunas de ellas a continuación.

2.1.8.1 Jetson AGX Xavier

El nuevo módulo Jetson AGX Xavier hace posibles las máquinas autónomas impulsadas por IA, que funcionan con tan solo 10W y entregan hasta 32 TOP. Como parte de la plataforma informática de inteligencia artificial líder en el mundo, se beneficia del amplio conjunto de herramientas y flujos de trabajo de inteligencia artificial de NVIDIA, que permiten a los desarrolladores capacitar e implementar redes neuronales rápidamente. Jetson AGX Xavier es compatible con NVIDIA JetPack SDK, que puede ayudarlo a ahorrar mucho al reducir el esfuerzo y los gastos de desarrollo. Como se observa en la Fig. 12 [8]

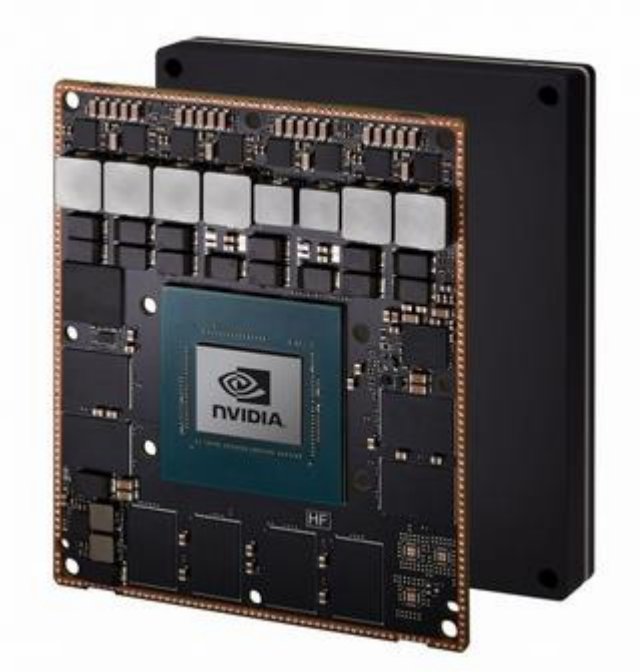

Fig. 12 Jetson AGX Xavier [8]

Tabla 1 ESPECIFICACIONES TECNICAS JETSON XAVIER AGX [8]

AI Performance	32 TOP
GPU	512-core Volta GPU with 64 Tenso Cores 11 TFLOPS (FP16) 22 TOPS (INT8)
CPU	8-core Carmel ARM v8.2 64-bit CPU, 8MB L2 + 4MB L3
Memory	32GB 256-Bit LPDDR4x | 136.5GB/s
Storage	32GB eMMC 5.1
DL Accelerator	(2x) NVDLA Engines* | 5 TFLOPS (FP16), 10 TOPS (INT8)
Vision Accelerator	7-way VLIW Vision Processor*
Video Encode	2x1000MP/sec 4x4K @60(HEVC) 8x4K @30(HEVC)

	16x1080p @60(HEVC) 32x1080p @30(HEVC)
Video Decode	2x1500MP/sec 2x8K @30(HEVC) 6x4K @60(HEVC) 12x4K @30(HEVC) 26x1080p @60(HEVC) 52x1080p @30(HEVC) 30x1080p @30(H.264)
Size	105 mm x 105 mm

2.1.8.2 Jetson Xavier NX

Jetson Xavier NX ofrece hasta 21 TOPS para ejecutar cargas de trabajo de IA modernas, consume tan solo 10 vatios de energía y tiene un factor de forma compacto más pequeño que una tarjeta de crédito. Puede ejecutar redes neuronales modernas en paralelo y procesar datos de varios sensores de alta resolución, lo que abre la puerta a dispositivos informáticos integrados y de borde que exigen un mayor rendimiento, pero están limitados por el tamaño, el peso y los presupuestos de energía. Como se visualiza en la Fig. 13 [8]

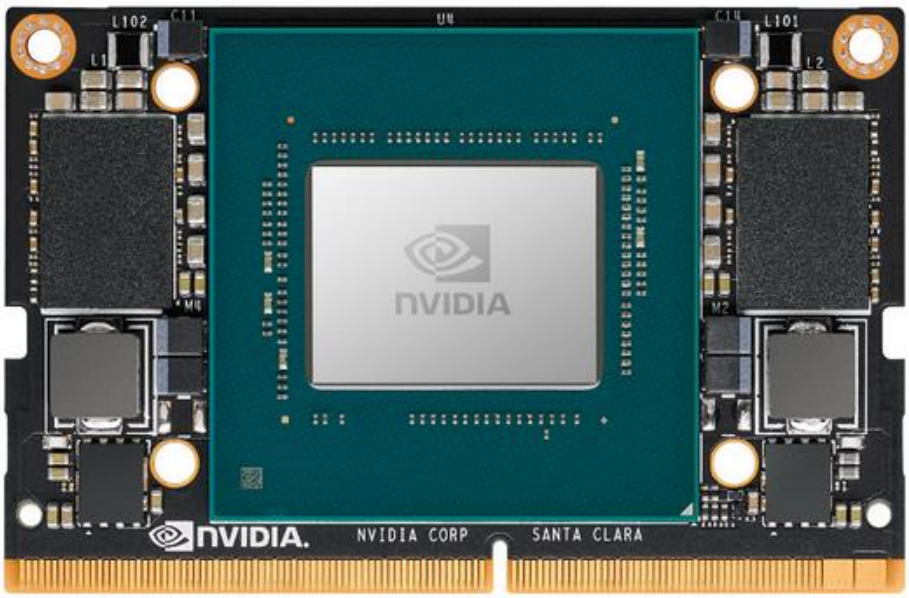

Fig. 13 Jetson Xavier NX [8]

Tabla 2 ESPECIFICACIONES TECNICAS JETSON XAVIER NX [8]

AI Performance	21 TOPS (INT8)
GPU	384-core NVIDIA Volta™ GPU with 48 Tensor Cores
GPU Max Freq	1100 MHz
CPU	6-core NVIDIA Carmel ARM®v8.2 64-bit CPU 6MB L2 + 4MB L3
CPU Max Freq	2-core @1900MHz 4/6-core @1400Mhz
Memory	8 GB 128-bit LPDDR4x @ 1866MHz 59.7GB/s
Storage	16 GB eMMC 5.1
Power	10W|15W|20W
PCIe	1 x1 + 1x4 (PCIe Gen3, Root Port & Endpoint)
CSI Camera	Up to 6 cameras (36 via virtual channels) 12 lanes MIPI CSI-2 D-PHY 1.2 (up to 30 Gbps)
Video Encode	2x 4K60 | 4x 4K30 | 10x 1080p60 | 22x 1080p30(H.265) 2x 4K60 | 4x 4K30 | 10x 1080p60 | 20x 108p30 (H.264)
Video Decode	2x 8K30 | 6x 4K60 | 12x 4K30 | 22x 1080p60 | 44x1080p30(H.265) 2x 4K60 | 6x 4K30 | 10x 1080p60 | 22x 1080p30 (H.264)
Display	2 multi-mode DP 1.4/eDP 1.4/HDMI 2.0
DL Acceleratot	2x NVDLA Engines

Vision Accelerator	7-Way VLIW Vision Processor
Networking	10/100/1000 BASE-T Ethernet
Mechanical	45 mm x 69.6 mm 260-pin SO-DIMM connector

2.1.8.3 Jetson Nano

Jetson Nano es una computadora pequeña y potente para aplicaciones integradas e IA IoT que ofrece el poder de la IA moderna.

Para su programación se utiliza el JetPack SDK con bibliotecas aceleradas para aprendizaje profundo, visión por computadora, gráficos, multimedia y más. Jetson Nano tiene el rendimiento y las capacidades que necesita para ejecutar cargas de trabajo de IA modernas, lo que le brinda una forma rápida y fácil de agregar IA avanzada a su próximo producto. Como se observa en la Fig. 14 [8]

Fig. 14 Jetson Nano [8]

Tabla 3 ESPECIFICACIONES TECNICAS JETSON NANO [8]

GPU	NVIDIA Maxwell architecture with 128 NVIDIA CUDA® cores
CPU	Quad-core ARM Cortex-A57 MPCore processor
Memory	4 GB 64-bit LPDDR4, 1600MHz 25.6 GB/s
Storage	16 GB eMMC 5.1

Video Encode	250MP/sec 1x4K @30(HEVC) 2x1080p @60(HEVC) 4x1080p @30(HEVC) 4x720p @60(HEVC) 9x720p @30(HEVC)
Video Decode	500MP/sec 1x4K @60(HEVC) 2x4K @30(HEVC) 4x1080p @60(HEVC) 8x1080p @30(HEVC) 9x 720p @60(HEVC)
Camera	12 lanes (3x4 or 4x2) MIPI CSI-2 D-PHY 1.1 (1.5 Gb/s per pair)
Connectivity	Gigabit Ethernet, M.2 Key E
Display	HDMI 2.0 and eDP 1.4
USB	4x USB 3.0, USB 2.0 Micro-B
Others	GPIO, I^2C, I^2S, SPI, UART
Mechanical	69.6mmx45mm 260-pin edge connector

2.1.9 GPIO

Entrada/Salida de propósito general (GPIO), es una señal digital a un pin dentro de la placa electrónica. Se puede encontrar en la Jetson Nano de NVIDIA una etapa de cabecera J41 en el cual se encuentran los pines GPIO. Estos pines se observan en la Fig. 15 [8]

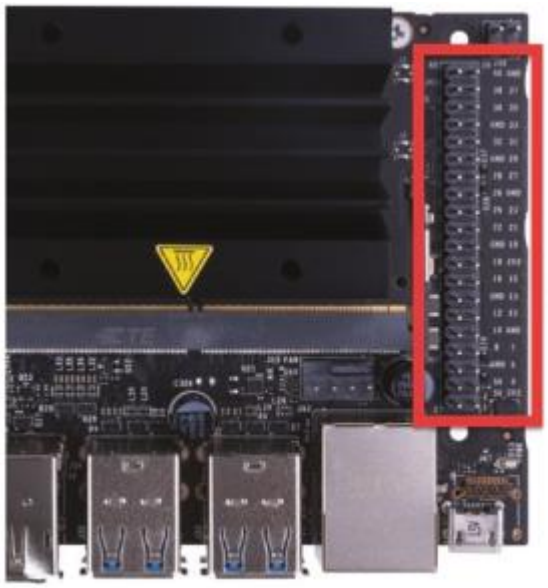

Fig. 15 Pines GPIO en la Jetson Nano NVIDIA [8]

El diagrama de pines de GPIO se puede observar en la Fig. 16. Aquí se puede visualizar los pines para GPIO, UART, SPI, I2S e I2C.

La tarjeta NVIDIA Jetson Nano GPIO pueden utilizar voltajes desde 1.8V hasta 3.3V. Por defecto, todos los pines GPIO utilizan 3.3V. Siempre se debe asegurar que el voltaje que ingrese a uno de los pines de entrada no supere los 3.3V porque dañaría la tarjeta electrónica Jetson Nano. [8]

Name	Pin	Pin	Name
3.3 VDC	1	2	5.0 VDC
I2C_2_SDA I2C Bus 1	3	4	5.0 VDC
I2C_2_SCL I2C Bus 1	5	6	GND
AUDIO_MCLK	7	8	UART_2_TX /dev/ttyTHS1
GND	9	10	UART_2_RX /dev/ttyTHS1
UART_2_RTS	11	12	I2S_4_SCLK
SPI_2_SCK	13	14	GND
LCD_TE	15	16	SPI_2_CS1
3.3 VDC	17	18	SPI_2_CS0
SPI_1_MOSI	19	20	GND
SPI_1_MISO	21	22	SPI_2_MISO
SPI_1_SCK	23	24	SPI_1_CS0
GND	25	26	SPI_1_CS1
I2C_1_SDA	27	28	I2C_1_SCL
CAM_AF_EN	29	30	GND
GPIO_PZ0	31	32	LCD_BL_PWM
GPIO_PE6	33	34	GND
I2S_4_LRCK	35	36	UART_2_CTS
SPI_2_MOSI	37	38	I2S_4_SDIN
GND	39	40	I2S_4_SDOUT

Fig. 16 Pines GPIO Jetson Nano NVIDIA [8]

Cada pin GPIO puede ser usado como pin de entrada o pin de salida. En la tarjeta Jetson Nano se puede observar cada uno de los pines etiquetados. Los números son muy útiles al momento de acceder a la GPIO de la tarjeta electrónica. [8]

2.1.10 Machine Learning

El aprendizaje automático, la aplicación y ciencia de algoritmos que hace sentido de los datos, es el campo de todas las ciencias de la computación. Se está viviendo en una época en la que abundan los datos; utilizando los algoritmos de auto aprendizaje del campo del aprendizaje automático, se puede convertir estos datos en conocimiento. Gracias a las bibliotecas de código abierto que se han desarrollado en los últimos años, probablemente nunca haya sido un mejor momento para entrar en el campo del aprendizaje automático y aprender cómo utilizar potentes algoritmos para detectar patrones en los datos y hacer predicciones sobre eventos futuros. [9]

Existen muchas aplicaciones, por ejemplo, de la visión al habla.

2.1.10.1 Construyendo máquinas inteligentes para transformar los datos en conocimientos

En esta era de la tecnología moderna, hay un recurso que se tiene en abundancia: por una parte, una gran cantidad de datos estructurados y, por otra parte, no estructurados. En la segunda mitad del vigésimo siglo, el aprendizaje automático evolucionó como un subcampo de la inteligencia artificial que involucraba el desarrollo de algoritmos de autoaprendizaje para obtener conocimiento de esos datos en para hacer predicciones. En lugar de requerir que los humanos deriven reglas manualmente y crear modelos a partir del análisis de grandes cantidades de datos, el aprendizaje automático ofrece una alternativa más eficiente para capturar el conocimiento en datos para mejorar gradualmente el rendimiento de modelos predictivos y tomar decisiones basadas en datos. No solo es el aprendizaje automático se vuelve cada vez más importante en la investigación en ciencias de la computación, pero también juega un papel cada vez más importante en nuestra vida diaria. Gracias al aprendizaje automático, disfrutamos de filtros de correo no deseado robustos, software conveniente de reconocimiento de voz y texto, motores de búsqueda confiables, ajedrecistas desafiantes y coches autónomos eficientes. [9]

2.1.10.2 Predicciones acerca del futuro con aprendizaje supervisado

El objetivo principal del aprendizaje supervisado es aprender un modelo a partir de datos de entrenamiento etiquetados, que permite hacer predicciones sobre datos futuros o no vistos. Aquí, el término supervisado se refiere a un conjunto de muestras donde las señales de salida deseadas (etiquetas) son ya conocidas. [9]

Considerando el ejemplo del filtrado de correo no deseado, se puede entrenar un modelo usando un algoritmo de aprendizaje automático supervisado en un corpus de correo electrónico etiquetado, correo electrónico que se correctamente marcado como spam o no spam, para predecir si un nuevo correo electrónico pertenece a cualquiera de las dos categorías. Una tarea de aprendizaje supervisada con etiquetas de clase discretas, como como en el ejemplo anterior de filtrado de correo no deseado, también se denomina tarea de clasificación. Otra subcategoría del aprendizaje supervisado es la regresión, donde la señal de resultado es un valor continuo: Se puede observar en la Fig. 17 [9]

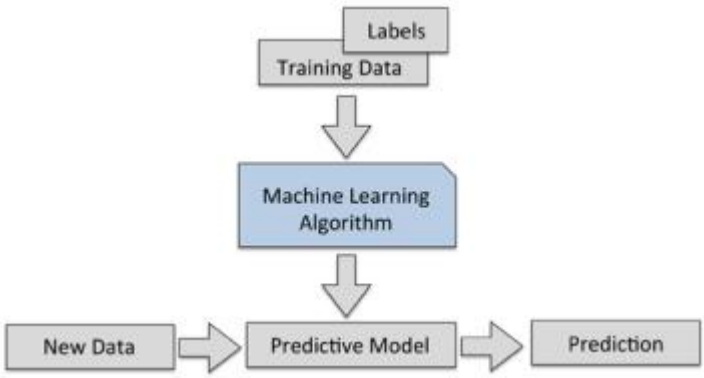

Fig. 17 Aprendizaje supervisado [9]

2.1.10.3 Sistema de aprendizaje automático

En esta sección, se discute otras partes importantes de un sistema de aprendizaje automático que acompaña al algoritmo de aprendizaje. El diagrama a continuación se muestra un diagrama de flujo de trabajo típico para usar el aprendizaje automático en modelado en la Fig. 18 [9]

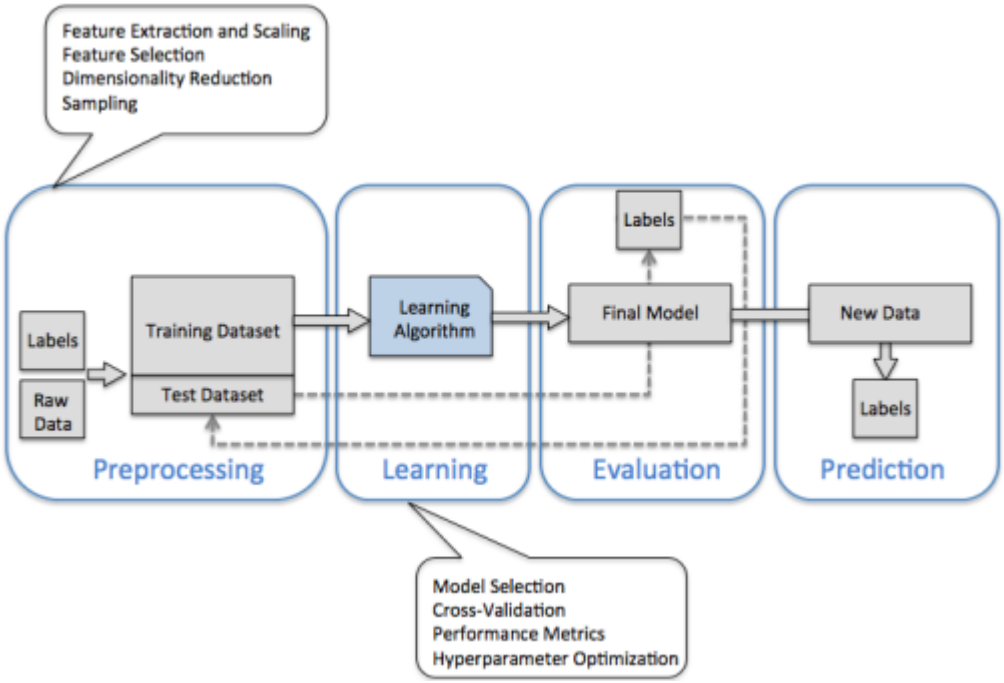

Fig. 18 Sistema de aprendizaje automático [9]

- **Preprocesamiento**

Los datos en bruto vienen con diferentes capturas de imágenes de varias posiciones para el óptimo rendimiento de un algoritmo de aprendizaje. Por tanto, el preprocesamiento de los datos es uno de los pasos más cruciales en cualquier aplicación de aprendizaje automático. Si el conjunto de datos sería de varios tipos de flores, podríamos pensar en los datos brutos como una serie de imágenes de flores de las que queremos extraer características significativas. Las características útiles pueden ser el color, la tonalidad, la intensidad de las flores, la altura, y las longitudes y anchos de las flores. En este caso se utilizará algoritmo de aprendizaje supervisado por lo que se requiere que las características seleccionadas estén en la misma escala para un rendimiento óptimo, que es a menudo se logra transformando las características en el rango [0, 1] o un estándar normal distribución con media cero y varianza unitaria.

Algunas de las características seleccionadas pueden estar altamente correlacionadas y, por lo tanto, ser redundantes. hasta cierto punto. En esos casos, las técnicas de reducción de dimensionalidad son útiles para comprimir las características en un subespacio de menor dimensión. Reducir la dimensionalidad del espacio de funciones

tiene la ventaja de que hay menos espacio de almacenamiento requerido, y el algoritmo de aprendizaje puede ejecutarse mucho más rápido. [9]

- **Algoritmo de entrenamiento**

Un punto importante que puede ser resumido de los famosos teoremas de no almuerzo gratis de David Wolpert es que no podemos aprender "gratis" (La falta de distinciones a priori entre algoritmos de aprendizaje, D.H. Wolpert 1996; No hay teoremas de almuerzo gratis para la optimización, D.H. Wolpert y W.G. Macready, 1997). Intuitivamente, se puede relacionar este concepto con el dicho popular "Yo

Suponga que es tentador, si la única herramienta que tiene es un martillo, tratar todo como si fuera eran un clavo. Por ejemplo, cada algoritmo de clasificación tiene sus sesgos inherentes, y ningún modelo de clasificación goza de superioridad si no haga suposiciones sobre la tarea. En la práctica, por tanto, es fundamental indicar un algoritmo de entrenamiento, en este caso es el algoritmo de aprendizaje de Microsoft ya que es una implementación de la popular y adaptable arquitectura de red neuronal para el aprendizaje automático. Uno de uso común métrica es la precisión de la clasificación, que se define como la proporción de instancias clasificadas. [9]

El conjunto de datos se divide en subconjuntos de entrenamiento y validación para estimar el rendimiento de generalización del modelo. Por último, tampoco se puede esperar el valor predeterminado. Los parámetros de los diferentes algoritmos de aprendizaje proporcionados por las bibliotecas de software son óptimo para la tarea específica. Por lo tanto, se hace un uso frecuente de técnicas de optimización de hiperparámetros que ayudan a ajustar el rendimiento de nuestro modelo. Intuitivamente, se puede pensar en esos hiperparámetros como parámetros que no se aprenden de los datos pero que representan las perillas de un modelo que se puede recurrir para mejorar su rendimiento. [22]

Evaluación de los modelos y predicción de instancias de datos no vistos

En el campo de aprendizaje profundo, una subcategoria, la cual se llama detección e objetos, la cual incluye una imagen, video o foto. A partir de la detección de objetos aparecieron modelo preentrenados como R-CNN que predijo desde 2014, basado en redes neuronales profundas (DNN), la detección de objetos ha ido creciendo con el tiempo por lo que empezaron aparecer nuevos modelos preentrenados basados en R-CNN como: SPP-NET, Fast_RCN, Faster RCNN y R-FCN. La precisión es alta, sin

embargo, la estructura de la red es bastante compleja. Esta evolución se puede observar en la Fig. 19 a continuación [22].

Fig. 19: Detección de objetos usando aprendizaje profundo [22]

Una vez que se han seleccionado un modelo que se ha ajustado al conjunto de datos de entrenamiento, se puede utilizar el conjunto de datos de prueba para estimar qué tan bien se desempeña en estos datos invisibles para estimar el error de generalización. Si su rendimiento llega a una estimación mayor al 80% de confianza, ahora se puede usar este modelo para predecir datos nuevos y futuros. Es importante tener en cuenta que los parámetros para los procedimientos mencionados anteriormente, como la escala de características y la reducción de dimensionalidad, se obtienen únicamente del conjunto de datos de entrenamiento, y los mismos parámetros posteriormente se vuelven para transformar el conjunto de datos de prueba, así como cualquier nueva muestra de datos: De lo contrario, el rendimiento medido en los datos de prueba puede ser demasiado optimista. [9]. Por lo cual, Mobilenet y redes neuronales binarias están dentro de las técnicas más modernas para modelos de aprendizaje profundo, esto complementa varias tareas en sistemas embebidos. Mobilenet Single Shot Multi-Box Detector (SSD) es una técnica para identificar un objeto considerando el aprendizaje profundo. Eta clase de algoritmo se utiliza para la detección de objetos en tiempo-real y para la alimentación de la cámara que se vaya utilizar, la cual detecta objetos en función a la transmisión de video. Asi que, se combina Mobilenet y SSD para que el métodosea rápido y eficiente basado en el aprendizaje profundo de detección de objetos. Este modelo de aprendizaje preentrenado muestra que la precisión promedio es de 99,76%, 97,76% y 71,07% para detectar diferentes clases de coche, persona y silla, respectivamente.[22]

2.1.11 Código QR

Cuando se refiere a un código QR (código de respuesta rápida) se lo puede identificar como un tipo de código de barras matricial o código bidimensional, estos códigos fueron diseñados para ser leidos por dispositivos como teléfonos inteligentes, ya que el código está formado por módulos de color negro puestos en un patrón cuadrado sobre un fondo blanco. La información que esta codificada puede ser una dirección URL, texto u otros datos, esto fué creado por la subsidiaria de Toyota Denso Wave en el año de 1994, el motivo por el cual fué diseñado el código QR es permitir que su contenido se decodifique a alta velocidad y la popularidad va creciendo en todo el mundo, sobre todo en Corea, EE.UU. y Japón. Al inicio se utilizaban para rastrear piezas en la fabricación de vehículos, pero actualmente su contexto se ha ampliado para la parte comercial, entretenimiento y dentro del marketing para el etiquetado de productos en perchas de micro mercados. [10]

Además, se pueden generar e imprimir sus propios códigos QR con el fin que otros lo puedan escanear para visitar varios sitios o aplicaciones de generación de códigos QR gratuitos y de pago. Por lo que Google ofrece un servicio de interfaz de programa para generar códigos QR y de igual manera para escanear y estas aplicaciones se puede instalar en casi todos los dispositivos de teléfonos inteligentes.[10]

2.1.11.1 Definición y tendencias actuales de los códigos QR

La función de los códigos QR es almacenar direcciones y localizadores uniformes para recursos que tienen la posibilidad de aparecer en tarjetas de presentación, autobuses, revistas o cualquier objeto en el cual los clientes o usuarios quieran adquirir información como podemos observar en la Fig. 20 [10]

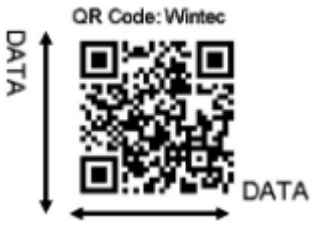

Fig. 20 Código QR [10]

Los clientes o usuarios que tengan un teléfono inteligente con cámara equipada y la correcta aplicación de escaneo de imagen para códigos QR, esto permitirá mostrar

información como texto, información de contacto, conectarse a una red inalámbrica o abrir una página web en el explorador o navegador del dispositivo móvil. También los códigos QR permiten interactividad. Los usuarios escanean el código con el fin de anunciar publicidad en sus dispositivos móviles, esto hace que aumente el potencial de interactividad haciendo clic en varios dispositivos o realizando una tarea específica para enlazar una oferta especial. Esto hace que los consumidores tengan una conexión con la marca. Por otro lado, los códigos QR se han hecho para tener un acceso más eficiente, rápido y práctico al contenido de teléfonos inteligentes. Por ejemplo, si se colocaría una función más compleja, el startup Pingtag los utiliza como una tarjeta de presentación digital para compartir cuentas de Linkedin e información de contacto. Otro ejemplo sería cuando Android implementa códigos QR para vincular directamente a Marketplace para descargar aplicaciones móviles. Otra aplicación sería para rastrear los parquímetros y también adquirir información de la Fundación del Patrimonio Mundial y de esta manera guiar a los visitantes a las tiendas cercanas o al estacionamiento, estas ubicaciones los lleva a través de Google maps. A la par Google está usando códigos QR para promocionar los distintos emprendimientos o negocios mediante google business.[10]

2.1.11.2 Comportamientos del usuario frente al código QR

Posiblemente una de las claves principales en comprender es el comportamiento de los usuarios en relación a los códigos QR, debido a que la industria podría mejorar campañas publicitarias que se adapten con mayor preferencia del cliente, en pocas palabras, la clave para el éxito del marketing de códigos QR es saber qué es lo que quiere el usuario. Es de suma importancia crear un modelo adecuado para reflejar la naturaleza de los códigos QR porque el modelos e encargaría de una evolución progresiva lo cual volvería más participativo, interactivo y de fácil acceso.

Además, cualquier tecnología implementada, esta debe ser percibida como útil para que sea asimilada y aceptada en las rutinas diarias. Los consumidores carecen de la sofisticación tecnológica para entender los códigos QR, por lo que deber ser fácil de escanear y comprender.[10]

2.1.11.3 Código QR y código de barras

En el medio para la identificación de objetos, existen las tecnologías RFID1 y NFC2 que esta consideradas de última generación, pero los códigos de barras todavía se consideran una opción interesante, debido a la tecnología básica y la simplicidad del

concepto. De hecho, se puede utilizar cualquier impresora económica, pero la creación de una RFID requiere un dispositivo dedicado especial. Estudios previos muestran la relevancia de usar códigos de barras con teléfonos equipados con cámara. Se estudiaron aplicaciones prototípicas que vinculan el mundo real. [11]

Los códigos de barras 1D clásicos son muy populares y universalmente reconocidos debido a su velocidad de lectura, precisión y características funcionales. Sin embargo, la necesidad de almacenar más información en códigos impresos en espacios reducidos conduce a la aparición de códigos de barras bidimensionales (Códigos 2D). Se pueden utilizar para acceder a datos y servicios como horarios de autobuses, información de productos, etc. La Fig. 21 muestra la misma información codificada en un código QR (izquierda) y en un código de barras EAN-13 (numeración europea de artículos) (derecha).[11]

Fig. 21 Código QR (2D) y Código de barras (1D) [11]

2.1.11.4 Tecnología código QR

Un código QR es un código matricial desarrollado y publicado principalmente para ser un símbolo que el equipo de escaneo puede interpretar fácilmente. Contiene información tanto en dirección vertical como horizontal, mientras que un código de barras clásico tiene solo una dirección de datos (generalmente la vertical). En comparación con un código de barras 1D, un código QR puede contener un volumen de información considerablemente mayor: 7.089 caracteres solo numéricos, 4.296 caracteres para datos alfanuméricos, 2.953 bytes de binarios (8 bits) y 1.817 caracteres de símbolos japoneses Kanji / Kana. El código QR también tiene capacidad de corrección de errores. Los datos se pueden restaurar incluso cuando partes sustanciales del código están distorsionadas o dañadas. En el estándar QR Code, las esquinas están marcadas y estimadas para que se pueda escanear el código interno [6]. El proceso de reconocimiento de códigos de barras consta de 5 pasos: (1) detección de bordes, (2) detección de formas, (3) identificación de la barra de control del código de barras, (4) identificación de la orientación, dimensiones y densidad de bits del código de barras mediante la barra de control y (5) calculando el valor del código de barras. [12]

2.1.11.5 Uso código QR

Para la decodificación manualmente de los códigos QR por parte del ser humano es imposible por lo cual es necesario una máquina que procese fácilmente mediante un escaneo, actualmente ya existen dispositivos con un software de lectura de códigos QR. Este software funciona de la siguiente manera: Mediante una fotografía del código QR, el software integrado en sus teléfonos decodifica los mensajes y muestra, manipula o almacena la información en sus dispositivos móviles. Dependiendo del tipo de datos reconocidos y la naturaleza de la aplicación, las acciones alternativas pueden seguir la etapa de decodificación: un número de teléfono se puede marcar automáticamente, se puede enviar un mensaje de texto corto, una página web correspondiente a la URL decodificada (Uniform Resource Locator) se puede mostrar en un navegador móvil, o se puede ejecutar una aplicación definida. Los códigos QR son parte de la vida diaria en Japón, Corea, Taiwán, Hong Kong y China. [13]

Un estudio publicado en enero de 2005 por MRI (Imagen de Resonancia Magnética) mostró que de 2053 usuarios de teléfonos móviles japoneses, el 90% ha reconocido un código QR. McDonalds usa códigos para informar a los usuarios sobre el valor nutritivo de sus hamburguesas. Apple anunció el nuevo i-Pod en vallas publicitarias con códigos QR. Los códigos QR utilizados en una campaña publicitaria de Nike permiten el acceso directo a un sitio móvil dedicado3. En Japón, algunos profesores utilizan códigos QR para distribuir recursos a los alumnos. Los códigos QR ahora aparecen en revistas, anuncios, envoltorios de productos, camisetas, pasaportes, tarjetas de visita y vallas publicitarias del metro en Japón. Pero, a nivel de mercado de consumidores, los códigos QR son prácticamente desconocidos fuera de Asia. Se cree que el uso de códigos QR en el resto del mundo no cobrará impulso hasta que los operadores de telefonía móvil comiencen a preinstalar el software de lectura de códigos QR en los teléfonos móviles. En la actualidad, en Europa son pocos los teléfonos que vienen con el software instalado. Sin embargo, varias empresas, principalmente en Francia, Reino Unido y Suiza, están empezando a utilizar códigos QR para promocionar bienes o servicios, como en el periódico online suizo, o para informar a los clientes sobre tarifas diarias, por ejemplo (Banco Central Europeo) [13].

2.1.11.6 Desarrolladores código QR

Los usuarios pueden escanear códigos QR existentes o pueden generar los suyos propios. Crear códigos QR en línea es muy fácil y se pueden utilizar muchos sitios web

(Kaywa, Snapmaze, Activeprint ...) para codificar e imprimir dichos códigos. Para teléfonos con cámara y PDA (Asistente digital personal) que no están equipados con lectores de códigos QR, existen algunas herramientas complementarias que decodifican los códigos QR simplemente colocando el dispositivo frente al código. Esto se hace automáticamente dentro del flujo de transmisión y el usuario no tiene que tomar una foto del código QR. Los lectores QuickMark [8] e I-nigma [9] son buenos ejemplos de herramientas gratuitas que utilizan esta técnica y que están disponibles para muchos modelos y dispositivos fabricados. QuickMark proporciona funcionalidades de extensión a los códigos QR, al permitir el cifrado total o parcial de los códigos. [13]

CAPÍTULO III

MARCO METODOLÓGICO

3.1 Ubicación

El presente proyecto de titulación se lo realizó en la provincia de Tungurahua, en la Parroquia de Totoras la cual pertenece a la ciudad de Ambato para cubrir las necesidades en la producción de molduras para cuadros, por lo cual se ha hecho el estudio para innovar y mejorar los tiempos de medición y etiquetado de los diferentes tipos de molduras.

3.2 Equipos y materiales

ID	EQUIPOS	PRECIO
1	Computadora	$800,00
2	Kit desarrollador Jetson Nano NVIDIA	$250,00
3	Mouse y teclado	$25,00
4	Cable HDMI	$12,00
5	Pantalla Intouch	$150,00
6	1 sensor ultrasónico	$50,00
7	1 Protoboard	$12,50
8	Cables Macho-Macho	$3,50
9	Caja impresa en 3D	$10,00
10	1 Arduino ATMega2560	$16,00
11	1 Cargador 9[v]	$12,50
	TOTAL	$1841,50

3.3 Tipo de investigación

Para la elaboración del presente proyecto se verá la necesidad de utilizar investigación de tipo documental, con el propósito de recopilar información de relevancia acerca de machine learning para implementar un sistema de control y etiquetado de molduras para cuadros en investigación basada en tutoriales, libros, revistas, científicas, páginas web e información teórica.

Además, se deberá utilizar la investigación de tipo aplicada, la cual nos permitirá analizar cada etapa del proceso del sistema electrónico para saber que función cumple cada subsistema desde las señales de entrada hasta las señales de salida del sistema de aprendizaje automático.

Otro de los puntos muy importantes es la investigación de tipo experimental, la cual permitirá analizar y conocer los diferentes tipos de modelos de entrenamiento para lo que son redes neuronales con el fin de obtener un sistema versátil, robusto y seguro, llegando a obtener datos del experimento y finalmente obtener los resultados por escrito.

3.4 Prueba de Hipótesis – pregunta científica - idea a defender

Implementar un sistema electrónico de control y etiquetado de molduras mediante machine learning para mejorar los tiempos de producción.

3.5 Recolección de información

Al momento de realizar la recolección de información nos basaremos en videotecas, revistas científicas, páginas web, datasheets, etc

3.6 Procesamiento de la información y análisis estadístico

Para la realización del procesamiento y análisis de datos se tomarán en cuenta las ssguientes actividades:

- Análisis de la información, la cual ayudará a formar y plantear estrategias con el fin de solucionar el problema.
- Interpretación de los resultados obtenidos, esto quiere decir la relación entre las variables más importantes que denotan la investigación del proyecto y generar una solución a partir del mismo.
- Revisión de la información indispensable para el proyecto, según las variables necesarias e importantes que se tomarán en cuenta.

3.7 Variables respuesta o resultados alcanzados

La implementación del sistema electrónico de control y etiquetado de molduras para cuadros en tiempo-real mediante machine learning permite mejorar los tiempos de producción en la fabricación de modelos de molduras, una vez que ya se tiene confirmado de forma verbal, se realiza las respectivas pruebas de funcionamiento del sistema completo. Después se realizarán los respectivos ajustes finales hasta cumplir los requerimientos necesarios en este capítulo.

Por lo cual se realizarón las siguientes pruebas para cumplir con el funcionamiento del sistema propuesto de la siguiente manera:

- Pruebas de funcionamiento de las distintas librerías que ofrece Python.
- Pruebas de funcionamiento de la cámara CSI de Raspberry la cual es compatible con la Jetson nano de NVIDIA.
- Pruebas de entrenamiento de la red neuronal de los diferentes modelos.
- Pruebas de funcionamiento al convertir el modelo aprendido a la extensión ONNX.
- Pruebas de funcionamiento al realizar el testeo con diferentes modelos de molduras.
- Pruebas de funcionamiento del microcontrolador Arduino ATMega2560 con el sensor ultrasónico para calcular la distancia de la tira de moldura.
- Pruebas de funcionamiento de la comunicación serial de los datos obtenidos en el dispositivo Arduino Atmega 2560 para ser enviado a Python.
- Pruebas de funcionamiento al momento de enviar la moldura por la banda transportadora para obtener la medida y el modelo de la tira.
- Finalmente, las pruebas de funcionamiento al visualizar la información obtenida en un código QR.

CAPÍTULO IV

RESULTADOS Y DISCUSIÓN

4.1 Análisis técnico del dispositivo Jetson Nano para la etapa de aprendizaje de los distintos tipos de moldura.

La placa base de la Jetson Nano de la empresa NVIDIA, es una placa potente y pequeña computadora que permite realizar operaciones en paralelo de múltiples redes neuronales para distintas aplicaciones como clasificación de imágenes, detección de objetos, segmentación y procesamiento del habla. Una de las ventajas es que tiene un entorno de desarrollo integral (JetPack SDK) y bibliotecas desarrolladas para aplicaciones integradas, aprendizaje profundo, visión por computadora, IoT, multimedia y mucho más. La tarjeta Jetson Nano tiene un procesador compatible con GeForce (GPU) usando los mismos núcleos CUDA, este crea un entorno poderoso de desarrollo de aplicaciones. Por otro lado, la Jetson Nano está constituida por una arquitectura heterogénea CPU-GPU en que el sistema operativo puede ser inicializado por la CPU y al momento de realizar tareas complejas de aprendizaje automático la GPU ayuda acelerando estas tareas debido a que es compatible con CUDA. Además, la inteligencia artificial revela que este dispositivo tiene bajo consumo de energía al momento de ejecutar los algoritmos. A continuación, se aprecia la computadora Jetson Nano en la Fig. 22 con sus características técnicas en la Tabla 4. [14]

Fig. 22 Jetson Nano NVIDIA [14]

Tabla 4 CARACTERISTICAS TECNICAS JETSON NANO [14]

Performance	**Jetson Nano** 472 GFLOPS
CPU	Quad-Core ARM Cortex-A57 64-bit@ 1.42 GHz
GPU	NVIDIA Maxwellw/ 128CUDA cores@921 MHz
Memory	4GB LPDDR4@ 1600MHz, 25.6 GB/s
Networking	Gigabit Ethernet/ M.2 Key E
Display	HDMI2.0 and eDP 1.4
USB	4xUSB3.0, USB2.0 Micro-B
Other	40-pin GPIO

Video Encode	H.264/H.265 (4Kp30)
Video Decode	H.264/H.265 (4Kp60,2x 4Kp30)
Camera	MIPICSI port
Storage	16GB eMMC
Power under load	5W-10W

4.2 Análisis técnico del microcontrolador Arduino Mega2560 para la etapa de transmisión.

La placa electrónica Arduino Mega 2560 se seleccionó porque es una placa muy completa y no es muy costosa en relación al Arduino Nano o Arduino UNO por lo que este dipositivo electrónico está compuesto por 54 entradas y salidas digitales, de las cuales se puede utilizar 15 como salidas PWM. Además, tiene 4 puertos seriales por hardware (UART), un cristal de 16 MHz, conexión USB, de igual manera para la alimentación tiene para un cargador de barril que puede ser entre 7 – 12 [v], está compuesto por un conector ICSP y un botón de reset. También se ha optado por este tipo de placa debido a que tiene compatibilidad con diferentes sensores como es el sensor ultrasónico que vamos a utilizar en este sistema electrónico para la medición de la moldura. El microcontrolador de nuestra placa es Atmel 2560 16AU la cula es fabricada con tecnología CMOS de 8 bits y baja potencia, está basado en la arquitectura RISC mejorada de AVR, su velocidad de transmisión es de 1Mbps lo cual genera una optimización de consumo de energía. A continuación, se puede detallar los pines de alimentación.

Vin: Este pin es utilizado para el voltaje de entrada a la placa cuando se tiene la opción de utilizar una fuente de alimentación externa.

- 5V: En este pin es suministrado un voltaje de 5V desde un regulador de la placa. La placa puede ser alimentada de CC(7-12V), el conector USB (5V) o el pin VIN de la placa (7-12V). Muy importante saber que suministrar directo a través de los pines de 5V y 3.3V puede causar daños a la placa electrónica debido a que ya no está conectado al regulador de voltaje para su protección interna.

  3V3: Es una fuente de 3.3V la cual es generada por un regulador de la placa, su consumo es de 50mA.

- GND: Pines de tierra

  IOREF: En este pin se encuentra la referencia de voltaje con la que funciona el microcontrolador.

  A continuación se puede observar el dispositivo electrónico en la Fig. 23 y las características técnicas que se ha colocado en la Tabla 5.[15]

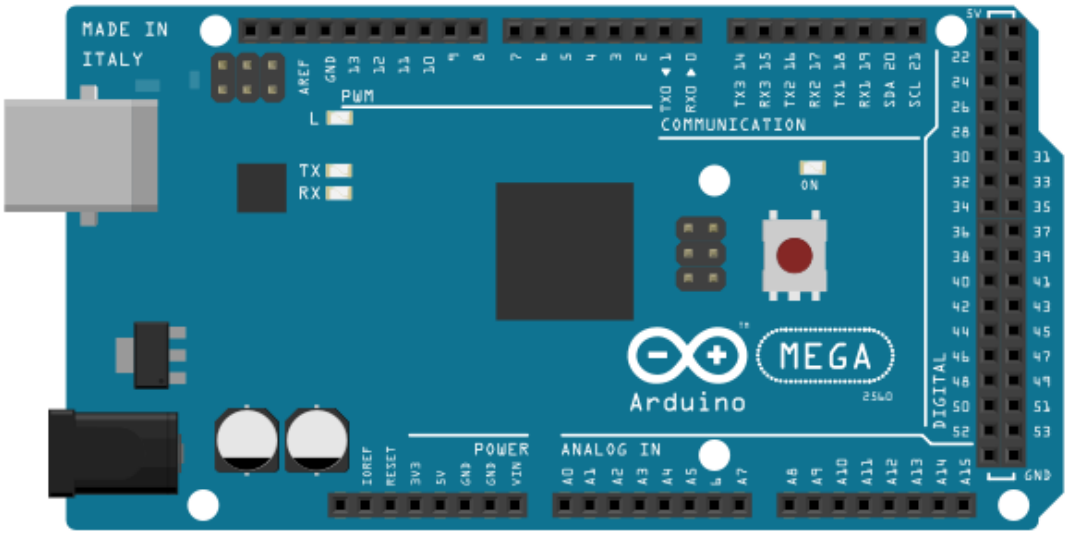

Fig. 23 Arduino Mega2560

Elaborado por: David Chávez

Tabla 5 CARACTERISTICAS TECNICAS ARDUINO MEGA 2560 [15]

Empresa	Arduino – ATMEL
Microcontrolador	ATMega 2560
Voltaje de operación	5V
Voltaje de alimentación (recomendado)	7-12V
Voltaje de alimentación (límites)	6-20V
Corriente DC en cada pin	40mA
Entradas/Salidas Digitales	54 (de los cuales 15 son PWM)
Entradas/Salidas Analógicas	16(solo entradas)
Memoria Flash	256 KB
EEPROM	4 KB
SRAM	8 KB
Rango de temperatura	-40°C a 85°C
Dimensiones	109 x 54 mm
Alimentación	Vía USB o fuente externa

4.3 Análisis técnico cámara raspberry pi V2 para la etapa de adquisición de imágenes mediante Jetson Nano NVIDIA

Al momento de seleccionar el tipo de cámara para conectarla a la tarjeta Jetson Nano de NVIDIA, se optó por la cámara raspberry pi debido a que tiene un sensor de imagen Sony IMX219 de alta calidad, esta tiene 8 megapíxeles diseñado con un lente de foco fijo. Tiene la capacidad de tomar imágenes estáticas de 3280x2464 pixeles, y también es compatible con video 1080p30, 720p60 y 640x480p90. Este tipo de cámara utiliza una interfaz dedicada a CSI. Este tipo de cámara lo podemos apreciar en la Fig. 24 y sus características técnicas en la Tabla 6.[16]

Fig. 24 Cámara Raspberry Pi V2 [16]

Tabla 6 CARACTERISTICAS TECNICAS CAMARA RASPBERRY PI V2 [16]

CARACTERISTICAS
8 megapíxeles de resolución nativa de alta calidad sensor de imagen Sony IMX219
Las cámaras son capaces de tomar imágenes estáticas 3280 x 2464 pixeles
Captura de vídeo a 1080p30, 720p60 y resoluciones 640x480p90
Todo el software está soportado dentro de la última versión del sistema operativo Raspbian
1.4 µmx 1,4 µm pixeles con la tecnología OmniBSI de alto rendimiento (alta sensibilidad, baja diafonía, bajo nivel de ruido)
Tamaño óptico de 1/4″
Dimensiones: 25mm x 23mm x 9mm / 0.98″ x 0.90″ x 0.35″
Peso (Cámara + cable): 3.4g

4.4 Características técnicas del sensor ultrasónico HC-SR04

El sensor seleccionado para detectar la distancia de la tira de moldura, se eligió el sensor ultrasónico HC-SR04 el cual es un circuito electrónico encargado de detectar o medir la distancia, este es compatible con Arduino Mega 2560. Además, este sensor detecta objetos, distancia o el nivel en el rango de 2cm hasta 400 cm. Se lo puede utilizar para implementar sistemas de alarmas de proximidad, medir niveles de aguas o cual objeto que almacene algún tipo de líquido. Es muy importante colocarlo de

manera perpendicular a la dirección de propagación del sensor. Este tipo de sensor se alimenta con 5V a 1.5mA. También trabaja con otros niveles lógicos, implementando un divisor de voltaje para ajustar el nivel. A continuación, se observa el sensor ultrasónico en la Fig. 25 y las características técnicas en la Tabla 7. [17]

Fig. 25 Sensor ultrasónico HC-SR04 [17]

Tabla 7 CARACTERISTICAS TECNICAS DEL SENSOR ULTRASONICO HC-SR04 [17]

Identificador	**Característica**
Alimentación	5 V
Interfaz de cuatro hilos	(VCC, trigger, echo, GND)
Rango de medición	2cm a 400cm
Corriente de alimentación	1.5mA
Frecuencia de pulso	40Khz
Apertura del pulso ultrasónico	15°
Señal de disparo	10us
Dimensiones del módulo	45x20x15mm

4.5 Requerimientos ubicación del sistema electrónico

El sistema electrónico de control y etiquetado de molduras para cuadros en tiempo-real mediante machine learning debe ir colocado paralelo de la banda transportadora de moldura porque:

1. Debe ser un lugar estratégico para ubicar la cámara raspberry pi V2 de manera correcta con el motivo de que pueda observar claro y preciso las tiras de moldura.
2. El sensor ultrasónico debe estar bien ubicado par evitar adquirir datos erróneos al momento de calcular la distancia de la tira de moldura.
3. El sistema no debe interrumpir o interferir con el funcionamiento correcto de la banda transportadora o el operador de la máquina.

4.6 Requerimientos técnicos de los equipos electrónicos del sistema

Cada uno de los dispositivos electrónicos que forman el sistema podemos detallarlo en la Tabla 8 a continuación:

Tabla 8 CONSUMO DE CORRIENTE DE LOS DISPOSITIVOS ELECTRONICOS DEL SISTEMA

Elaborado por: David Chávez

Dispositivo Electrónico	**Consumo de Corriente**
Arduino Mega2560	40mA
Jetson Nano NVIDIA	90mA
Sensor ultrasónico HC-SR04	1.5mA
Total	131.5mA

Fuente de alimentación Jetson Nano NVIDIA = 5V/4A

Fuente de alimentación Arduino Mega2560 = 9V/2A

4.7 Selección del entorno de programación del sistema de control

4.7.1 Python

Python es uno de los lenguajes de programación más utilizado de manera mundial, debido a que posee versatilidad, agilidad para el desarrollo ya que cuenta con un intérprete cuando se ejecuta el código. Es muy práctico y útil al momento de utilizar y probar fragmentos de código. Además, cuenta con un sin fin de herramientas para facilitar su entendimiento. Cuenta con multiplataforma y esto hace que permita que dicho lenguaje pueda ejecutarse en diferentes sistemas operativos y/o computadoras. Además, es uno de los lenguajes de programación más populares para la ciencia de datos, así de esta manera cuenta con útiles bibliotecas complementarias desarrolladas por su comunidad. A pesar de que el rendimiento de los diferentes lenguajes

interpretados por Python, para tareas de computación intensiva es inferior a los lenguajes de programación de nivel inferior. Por ejemplo, se tiene NumPy y SciPy que sirven para implementar operaciones rápidas y vectorizadas en matrices multidimensionales. Por otro lado, las tareas de programación de aprendizaje automático, se basa principalmente en la biblioteca Scikit-learn, esta es una de las máquinas de código abierto más populares y accesibles. También, Python funciona en los tres tipos de sistemas operativos como son: Microsoft Windows, Mac OS X y Linux. En este proyecto los paquetes principales de Python que se instalarán son los siguientes: [18]

Numpy

SciPy

Scikit-learn

Matplotlib

Pandas

Aquí se va a enfocar en el aprendizaje supervisado, el cual está compuesto por dos subcampos: Clasificación y regresión. Mientras los modelos de clasificación permiten categorizar objetos dentro de clases conocidas, se va poder predecir continuamente las etiquetas generadas. Frank Rosenblatt publicó el primer concepto del perceptrón, este es una regla de aprendizaje basado en el modelo de neuronas, este algoritmo permite que de manera automática aprenda coeficientes de peso óptimos que luego se multiplican con las características de entrada con el fin d etomar una decisión de si una neurona dispara o no. Este algoritmo se implementaba para predecir si una muestra pertenecía a una clase u otra. En este caso se podría representar como una clasificación binaria 1 (clase positiva) y -1 (clase negativa) y de esta manera se puede definir una función de activación $\varphi(z)$ donde se toma una combinación lineal debido a que los valores de **x** son las entradas y el vector **w** corresponden a sus pesos y de esta manera se forma la red de entrada ($z=w_1z_1+...+w_mz_m$). [18]

La plataforma que se está utilizando para programar en Python es PyCharm que a continuación se darán algunos detalles:

Al momento de empezar a programar en Python, se usará el IDE PyCharm debido a su versatilidad y su sencillez de descarga y actualización de las diferentes librerías que se va a utilizar durante el proceso de programación del sistema electrónico de control y etiquetado de molduras para cuadros en tiempo-real mediante machine learning. Por

otro lado, este IDE es un entorno integrado, debido a que ofrece multitud de herramientas de manera que no tiene que salirse de la aplicación para realizar cualquier tipo de tarea involucrada en su desarrollo. Aquí se puede involucrar inspectores de código, integración con base de datos, docker, sistemas de versionado de datos, herramientas de depuración, análisis de código, test, depuración en remoto. Se puede observar en la Fig. 26 El entorno gráfico de Pycharm. [19]

El trabajo de titulación se realizó en la plataforma de programación Pycharm 2021.3, esta plataforma es es instalada en el sistema operativo Ubuntu 20.04. Además, dentro de la plataforma Pycharm se instaló Python 3.8.

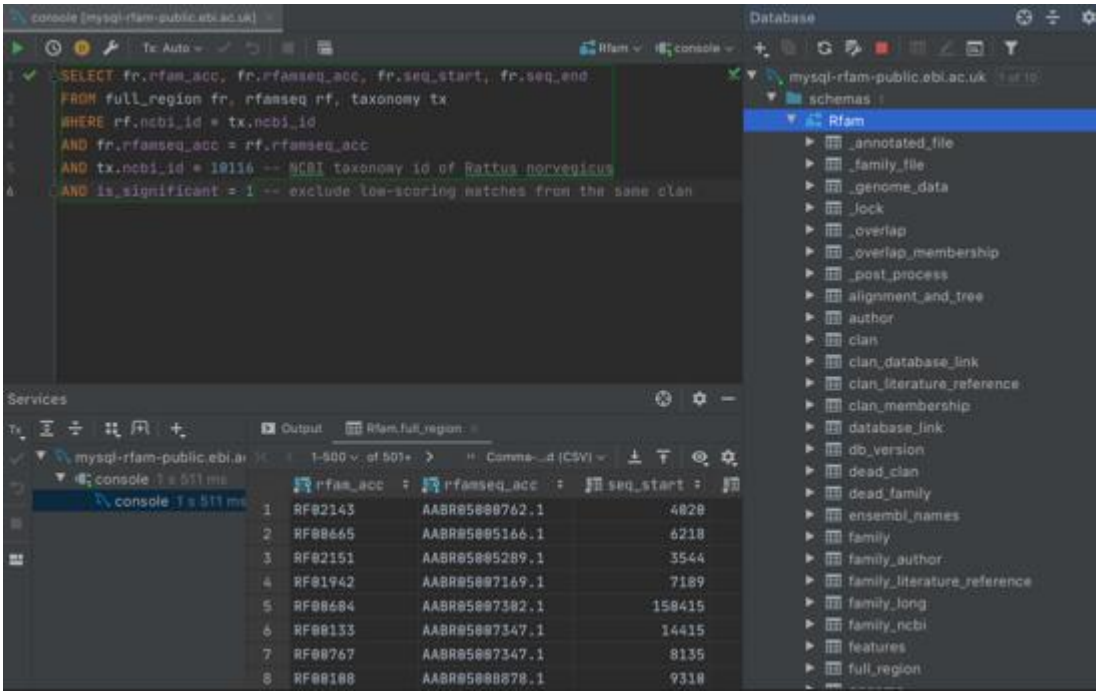

Fig. 26 Entorno gráfico de Pycharm [19]

4.7.2 Arduino

Arduino se lo utiliza como un hardware de código abierto, programable mediante una plataforma de software propio, donde se puede encontrar contenidos creados por una comunidad global. Esta plataforma facilita realizar proyectos interactivos, la cual es muy potente y es utilizada para desarrollar proyectos tanto en el campo de la ingeniería, la robótica, etc. Se puede observar en la Fig. 27 la interfaz gráfica de Arduino. [20]

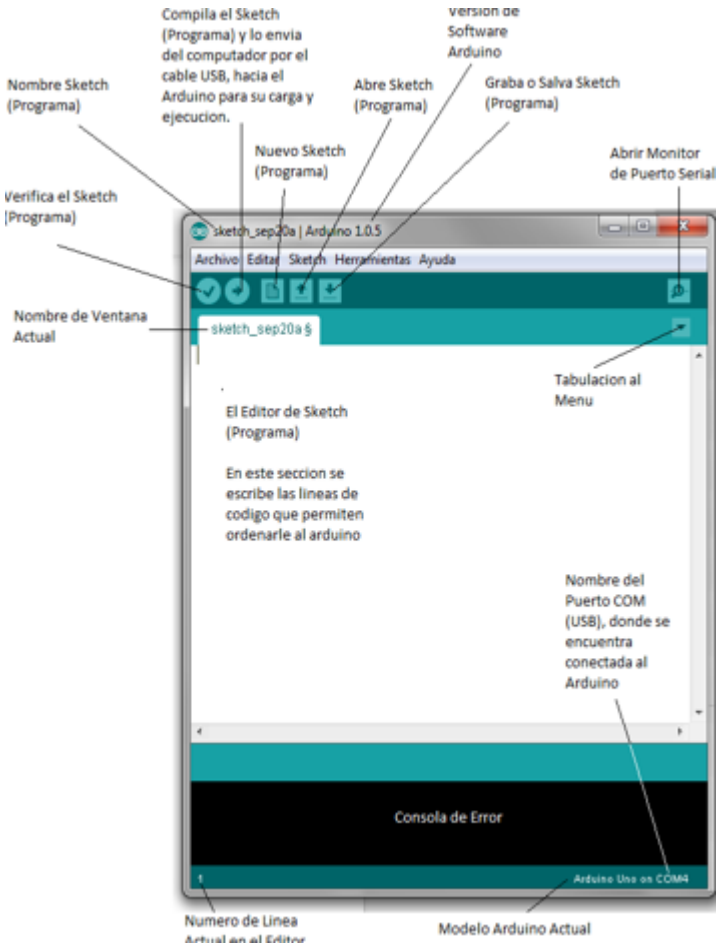

Fig. 27 Interfaz gráfica Arduino [20]

4.8 Diseño del sistema electrónico de control

El esquema general del sistema electrónico de control y etiquetado de molduras para cuadros en tiempo-real mediante machine learning se observa en la Fig. 28. está compuesto por:

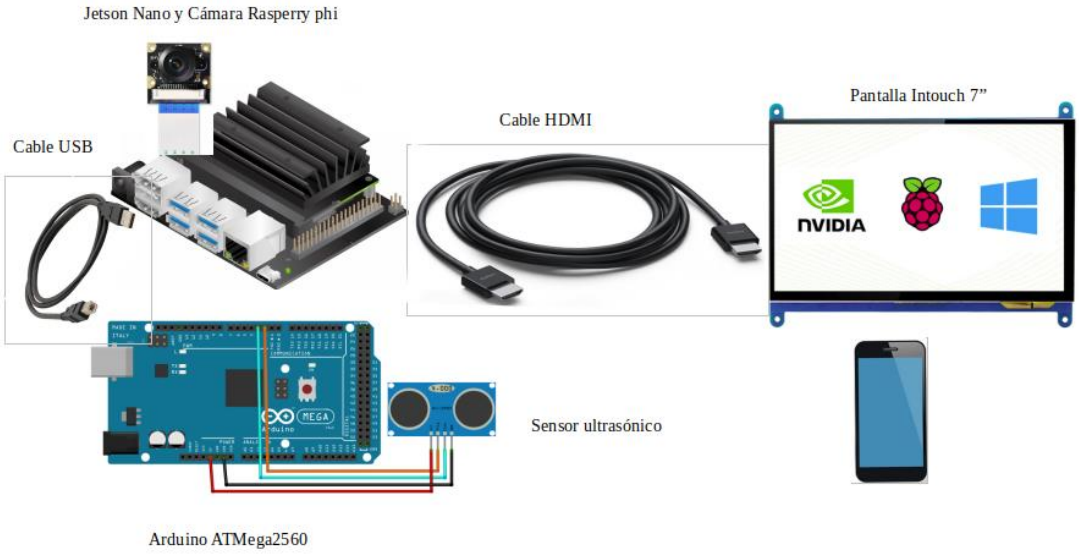

Fig. 28: Esquema general del sistema electrónico de control y etiquetado de molduras para cuadros en tiempo-real mediante machine learning
Elaborado por: David Chávez

4.8.1 Etapa de aprendizaje automatizado (molduras de cuadros)

- Jetson Nano NVIDIA
- Cámara Raspberry pi V2
- Pantalla Intouch 7¨

Para el desarrollo del proyecto el cual va detectar diferentes tipos de molduras mediante machine learning el cual se ha realizado un proceso de tratamiento de imágenes siguiendo diferentes etapas en el dispositivo electrónico Jetson Nano NVIDIA para conseguir la predicción del modelo de moldura que la cámara Raspberry pi este observando y poder visualizarlo en una pantalla Intouch 7¨. A continuación, se explica el sistema de detección de objetos (Molduras). La Fig. 29 se muestra el sistema para la detección de molduras. La tarjeta Jetson Nano NVIDIA es muy práctica y versátil para utilizar iferencia de aprendizaje profundo en funcionamiento para la detección de los diferentes tipos de molduras para cuadros en tiempo real, ya que se utiliza modelos previamente entrenados en unión de su JetPack SDK lo cual ayuda para la aceleración computacional al momento de trabajar con redes neuronales; por otra parte tenemos TensorRT que ayuda a la cuantificación y optimización al momento de implemetar aplicaciones de inferencia de aprendizaje profundo como transmisión de video de los distintos modelos de moldura, esto hace que la inferencia de precisión reduzca la latencia para mejorar la aplicación en tiemp-real.

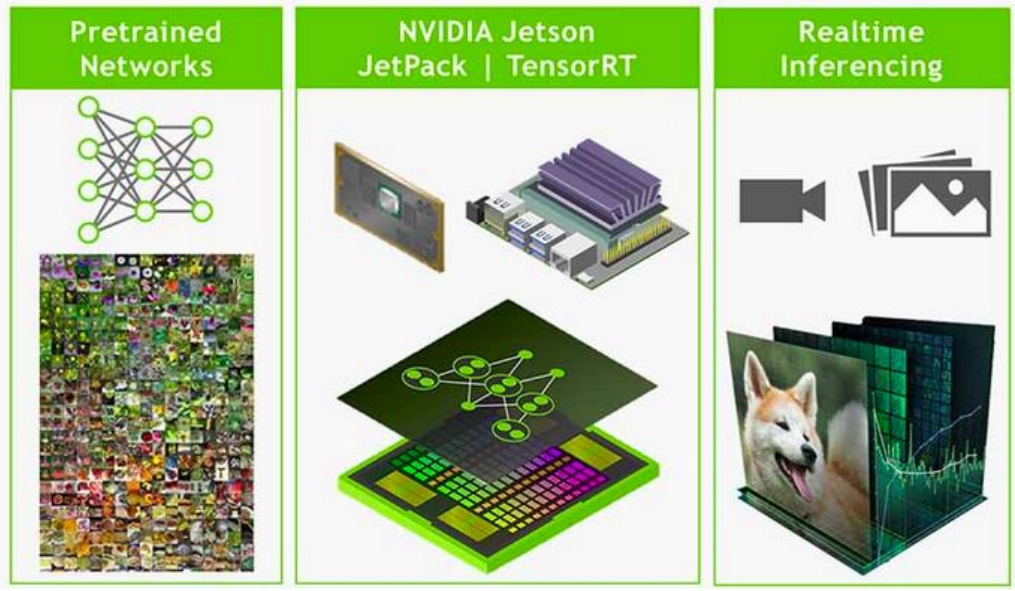

Fig. 29 Etapas para el entrenamiento de molduras [21]

4.8.1.1 Transferir aprendizaje con PyTorch

En el presente proyecto de detección del tipo de moldura se pretende utilizar el aprendizaje por transferencia debido a que es una técnica para volver a entrenar un modelo DNN (Redes Neuronales Profundas) en nuestro nuevo conjunto de datos, esto va a llevar a que el tiempo sea menos para su entrenamiento y evitar crear una red desde cero. Con este método, los pesos de un modelo previamente entrenado se ajustan para clasificar con el conjunto de datos. Se va a implementar la red llamada SSD-Mobilenet, también hay la posibilidad de utilizar otro tipo de redes, entre las cuales puede ser ResNet-18. Para lo cual se observa en la Fig. 30 [21]

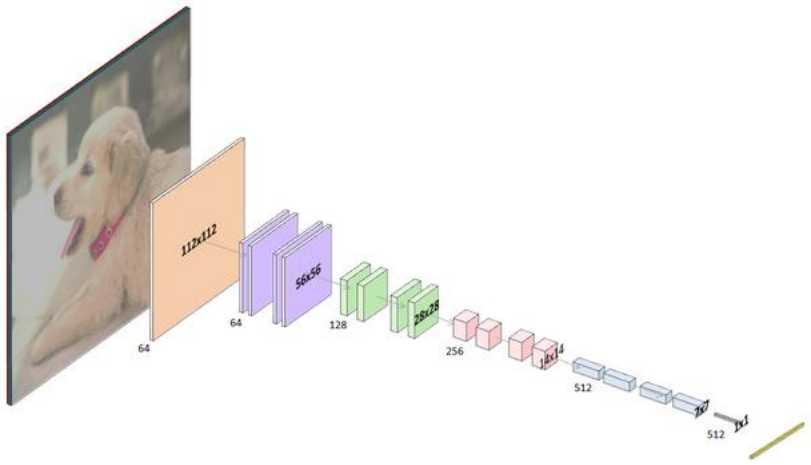

Fig. 30 Transferir aprendizaje con Pytorch [21]

Al momento de realizar la etapa de entrenamiento, generalmente se lo realiza en la PC, servidor o instancia con GPU discretas ya que utiliza un gran conjunto de datos y las

demandas computacionales son complementadas. A través del aprendizaje de transferencia se puede volver a entrenar varias redes a bordo de Jetson para los propios modelos DNN. [21]

PyTorch es un marco de aprendizaje automático que se usará en el presente proyecto, por lo cual se proporciona un conjunto de datos, ya que se va a utilizar herramientas como la cámara raspberry pi para recopilar y etiquetar este conjunto de datos de entrenamiento. [21]

4.8.1.2 Entrenamiento con SSD-Mobilenet

A continuación, se presenta la etapa de entrenamiento del modelo de detección de objetos SSD-Mobilenet para diferentes tipos de molduras mediante PyTorch y el conjunto de datos recopilados a través de la cámara raspberry pi. La arquitectura de red llamada SSD-Mobilenet, es utilizada para deteción de objetos en tiempo real en dispositivos móviles e integrados que combina el detector SSD-300 (single-shot multiBox) con una red troncal Mobilenet como se observa en la Fig. 31 [21]

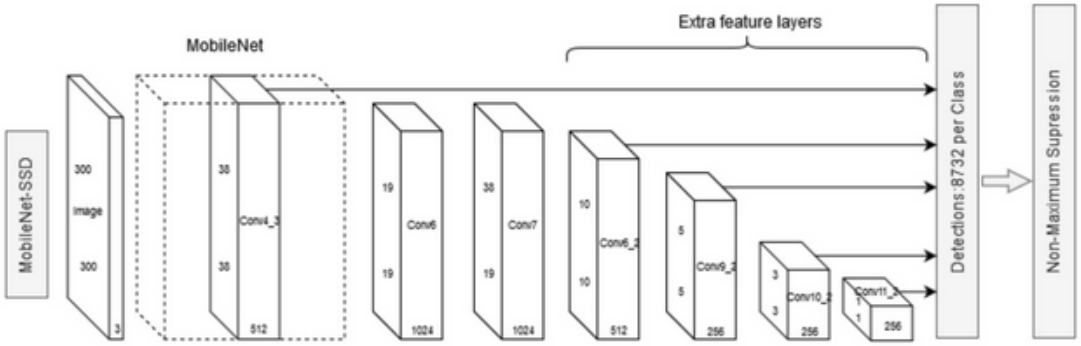

Fig. 31 Arquitectura SSD-Mobilenet [21]

El entrenamiento con la red SSD-Mobilenet tiene un estimado de 15.91 imagenes por segundo de un total de 2100 imágenes recopiladas, en la siguiente Tabla 9.

Tabla 9 TIEMPO DE ENTRENAMIENTO CON SSD-MOBILENET

	Imagenes/sec	**Tiempo por época**
Jetson Nano NVIDIA	15.91	2 min 12 segundos

1. Crear un archivo de etiquetas

Para lo cual se debe ingresar al directorio:
jetson-inference/python/training/detection/ssd/data, aquí se crea un directorio con el conjunto de datos de las molduras y un archivo de texto con los distintos modelos

de molduras, usualmente se crea un archivo de texto con el nombre labels.txt como se observa en la Fig. 32

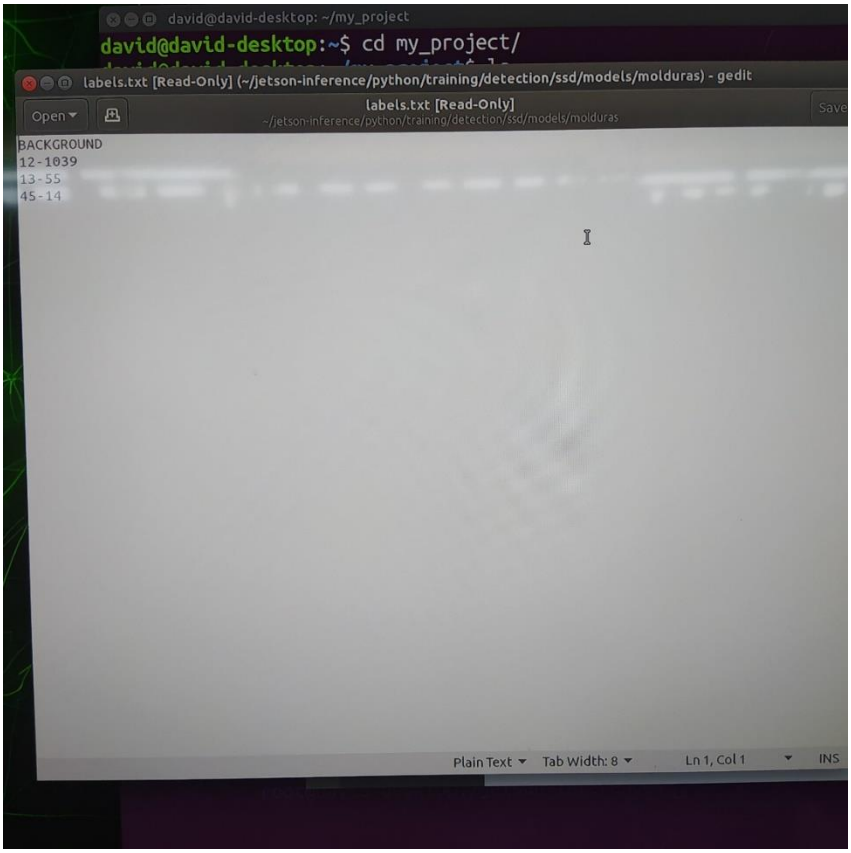

Fig.32 Archivo labels.txt de los modelos de molduras

Elaborado por: David Chávez

2. Recolectar un conjunto de datos

En esta etapa se realiza la captura de fotos de cada uno de los modelos de molduras que se quiera almacenar. Los modelos de moldura que se clasificarán son tres y estos son 12-1039, 13-55 y 45-14, así se puede visualizar respectivamente en la Fig. 33. Por lo cual se tiene la opción de una de las herramientas que el docker ofrece para tomar fotos con la cámara raspberry pi. Esta herramienta se llama **camera-capture** y permite capturar fotos en formato Pascal VOC como se observa en la Fig. 34, la ventana control de captura, después de que el menú desplegable tipo de conjunto de datos se haya configurado en **modo Detección**. [21]

$ camera-capture csi://0 # using default MIPI CSI cam$ camera-capture /dev/video0 # using V4L2 camera /dev/video0

Fig. 33: Modelos de moldura para entrenar Elaborado por: David Chávez

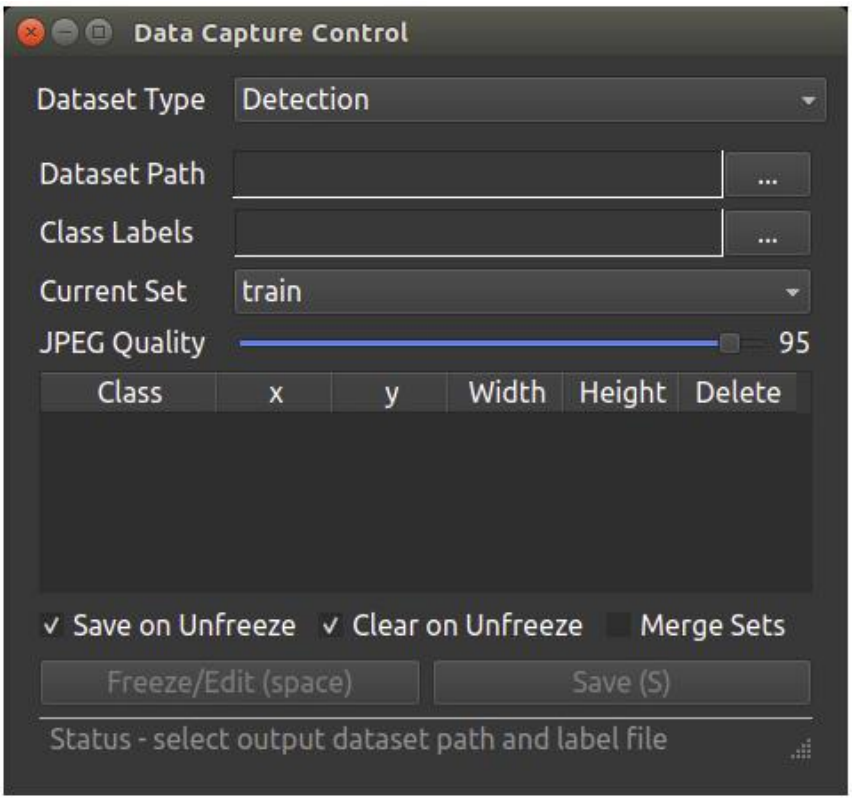

Fig. 34 Ventana Control de captura de fotos [21]

Se debe tomar en cuenta: si desea etiquetar un conjunto de imágenes que ya tiene (en lugar de capturarlas desde la cámara), se debe intentar usar una herramienta como **CVAT (Herramienta de anotación de visión artificial)** y exporte el conjunto de datos en formato Pascal VOC, el cual es un archivo XML que contiene las coordenadas de las imagenes capturadas. Como se puede observar en la Fig. 35 Luego cree un label.txt en el conjunto de datos con los nombres de cada una de sus clases de objetos. A continuación se puede observar la captura de fotos de diferentes modelos de molduras, en este caso se captura el modelo 13-55, se puede observar en la Fig. 36 [21]

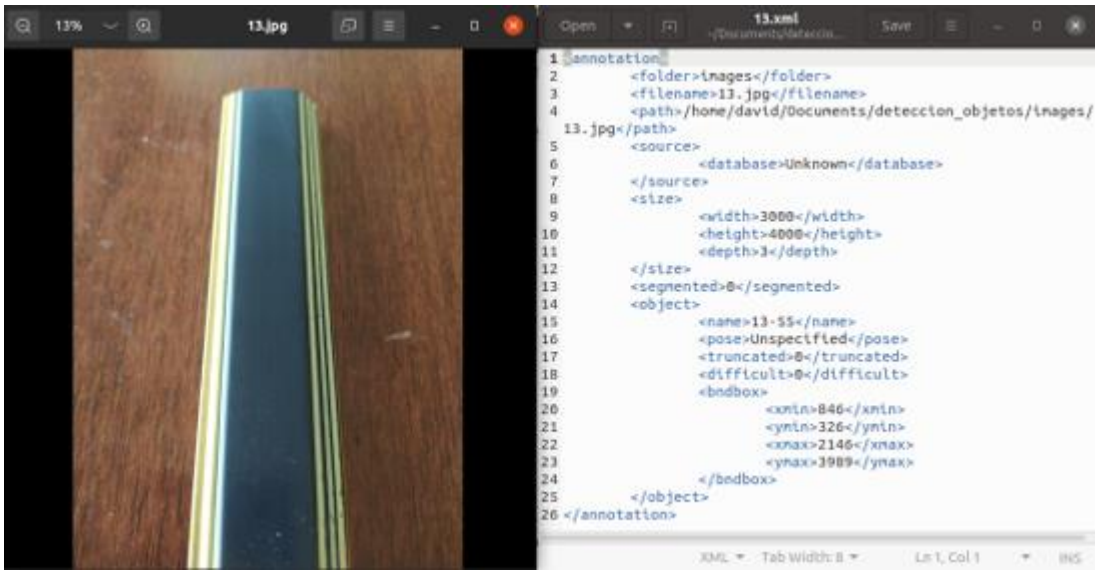

Fig. 35: Foto modelo de moldura 13-55 y la informción en archivo .xml
Elaborado por: David Chávez

Fig. 36 Captura de imágenes con la herramienta camera-capture

Elaborado por: David Chávez

La Fig. 37 se muestra como está formado la herramienta de captura de imágenes para formar el conjunto de datos de los diferentes modelos de molduras y a continuación se explica el funcionamiento de cada botón:

Luego, se abre la ruta del conjunto de datos y las etiquetas de clase que fueron creadas. Los botones Congelar/Editar y Guardar se activarán.

Se coloca la cámara en los objetos de la escena y se da clic en el botón Congelar/Editar (o se presiona la barra espaciadora). La vista de la cámara en vivo se 'congelará' y se podrá dibujar cuadros delimitadores sobre los objetos. A continuación, puede seleccionar la clase de objeto adecuada para cada cuadro delimitador en la tabla de cuadrícula en la ventana de control. Cuando se ha terminado de etiquetar la imagen, se vuelva a dar clic en el botón Congelar/Editar presionado para guardar los datos y descongelar la vista de la cámara para la siguiente imagen. [21]

Otros widgets en la ventana de control incluyen como se observa en la Fig. 34:

- Save on unfreeze: Automaticamente guarda los datos cuando Freeze/Edit este descongelado.
- Clear on Unfreeze: Automaticamente elimina los cuadros delimitadores anteriores al descongelar.
- Merge Sets: Guarda los mismos datos en train, val y test
- Current Set: Se selecciona de las carpetas train/val/test.

  Para la detección del tipo de moldura se necesita al menos tener imágenes almacenadas en train y test.

  Aunque si se marca Merge Sets, los datos se replicarán en train, val y test.
- **JPEG Quality:** Controlar la calidad de la codificación y el tamaño del disco de las imágenes guardadas

Es muy importante que los datos recopilados sean desde diferentes ángulos, puntos de vista de la cámara, orientaciones de objetos, condiciones de iluminación, diferentes fondos con el fin de crear un modelo resistente al ruido y muchos cambios en el entorno. En el caso de que el modelo de verificación de los distintos tipos de moldura no está funcionando bien es recomendable agregar más datos de entrenamiento y jugar con las condiciones. [21]

3. Entrenar el modelo

Después de recopilar una gran cantidad de datos, se puede empezar a entrenar el modelo con el script train_ssd.py. El proceso de entrenamiento utiliza ellas siguientes lineas de código que se va a utilizar en el terminal, en el presente caso se lo implemento en Ubuntu 20.04 de la siguiente manera: [21]

$ cd jetson-inference/python/training/detection/ssd

$ python3 train_ssd.py --dataset-type=voc --data=data/<YOUR-DATASET> --model-dir=models/<YOUR-MODEL>

A continuación se puede observar en la Fig. 36 el diseño del modelo de detección de molduras.Después de realizar la etapa de entrenamiento es muy importante convertir del modelo PyTorch a formato ONNX con la siguiente línea de código en la terminal.

$ python3 onnx_export.py –model-dir=models/<YOUR-MODEL>

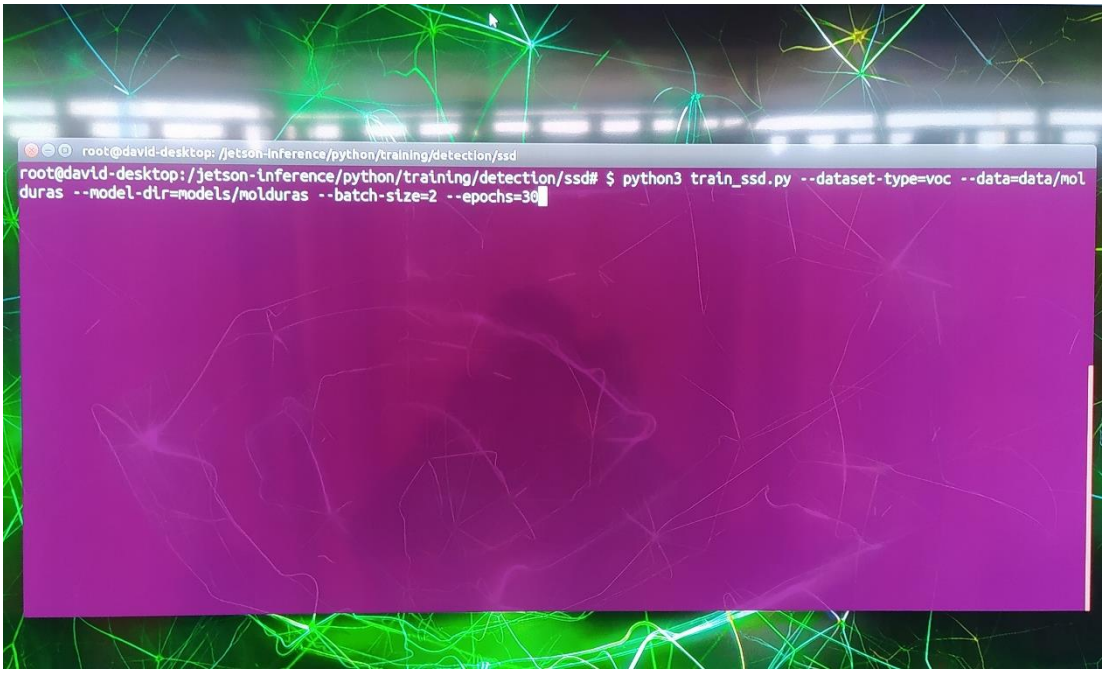

Fig. 37: Entrenamiento modelo molduras

Elaborado por: David Chávez

A continuación, se demuestra que se ha entrenado el modelo y ahora se la lleva al formato ONNX como se puede observar en la Fig. 37

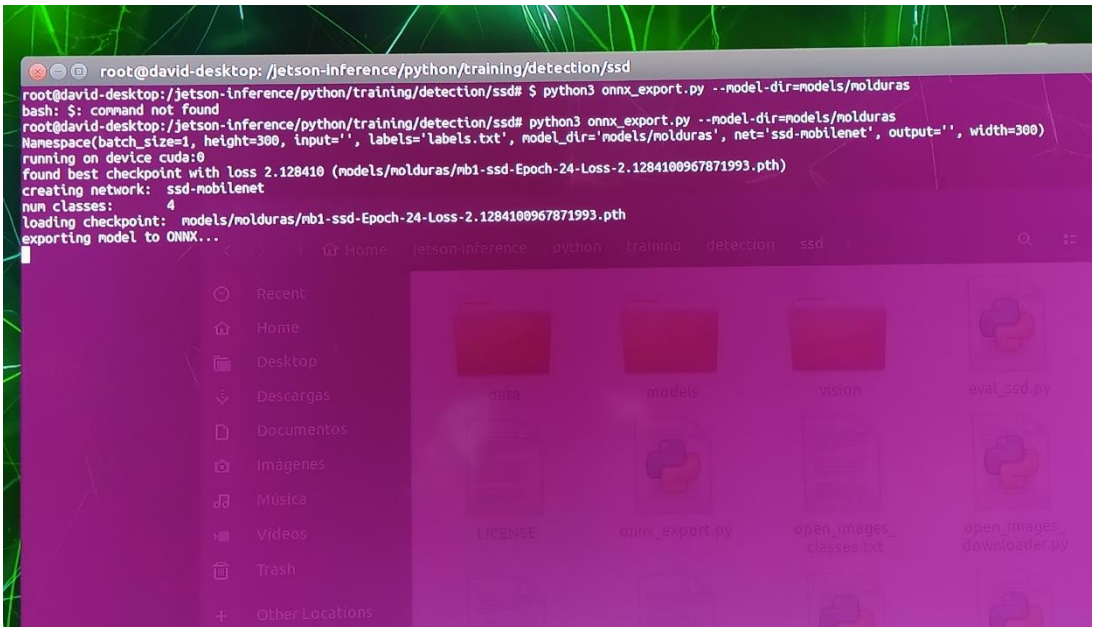

Fig. 38: Modelo convirtiendo a formato ONNX

Elaborado por: David Chávez

El modelo convertido será guardado bajo **<YOUR-MODEL>/ssd-mobilenet.onnx**, el cual se puede cargar con el programa **detectnet** que se presenta a continuación:

NET=models/<YOUR-MODEL>

detectnet --model=$NET/ssd-mobilenet.onnx –labels=$NET/labels.txt --input-blob=input_0 --output-cvg=scores --output-bbox=boxes csi://0

Si es necesario, regresar y recopilar más datos de entrenamiento para volver a entrenar el modelo. Puede reiniciar nuevamente y continuar donde lo dejó usando el argumento --resume (ejecute python3 train_ssd.py --help para obtener más información). Recuerde volver a exportar el modelo a ONNX después de volver a entrenar. [21]

4.8.2 Etapa de medición de los distintos modelos de moldura.

- Arduino Mega2560
- Sensor ultrasónico HC-SR04

En la etapa de medición de cualquier tipo de modelo de moldura se va a utilizar el microcontrolador Arduino Mega 2560 en donde se va programar el algoritmo para calcular la longitud de la tira de moldura mediante un sensor ultrasónico HC-SR04. La codificación del programa se la realiza en la interfaz Arduino IDE, en la Fig. 39 se muestra la conexión, realizada en el software Fritzing y en la Fig. 40 se muestra la conexión física.

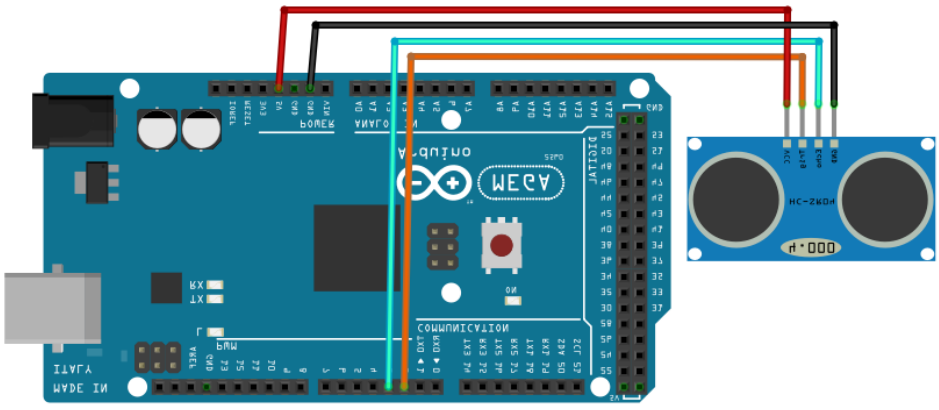

Fig. 39: Arduino Mega 2560 y Sensor Ultrasónico HC-SR04 hecho en Fritzing

Elaborado por: David Chávez

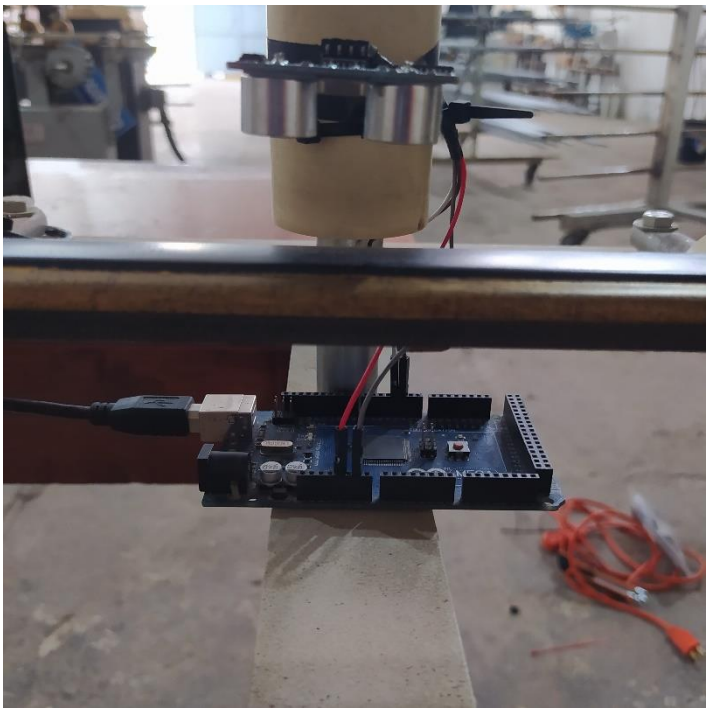

Fig. 40: Conexión física entre Arduino Mega 2560 y Sensor Ultrasónico HC-SR04
Elaborado por: David Chávez

4.8.3 Etapa de generación de códigos QR

En la penúltima etapa del sistema se ha implementado la generación de códigos QR con el objetivo de almacenar el tipo de modelo de moldura y la dimensión de la tira de moldura para optimizar los tiempos de etiquetado, esto ayudará en la producción en serie aprovechando la información y acelerando los procesos de inventario.

La programación basada en Python para almacenar toda la información en un solo programa. Como se indicó anteriormente, la etapa de medición de la tira de moldura se la ha realizado en el IDE de Arduino y mediante comunicación serial se lo ha llevado a Python. La etapa de generación de códigos QR se puede observar en la Fig. 41

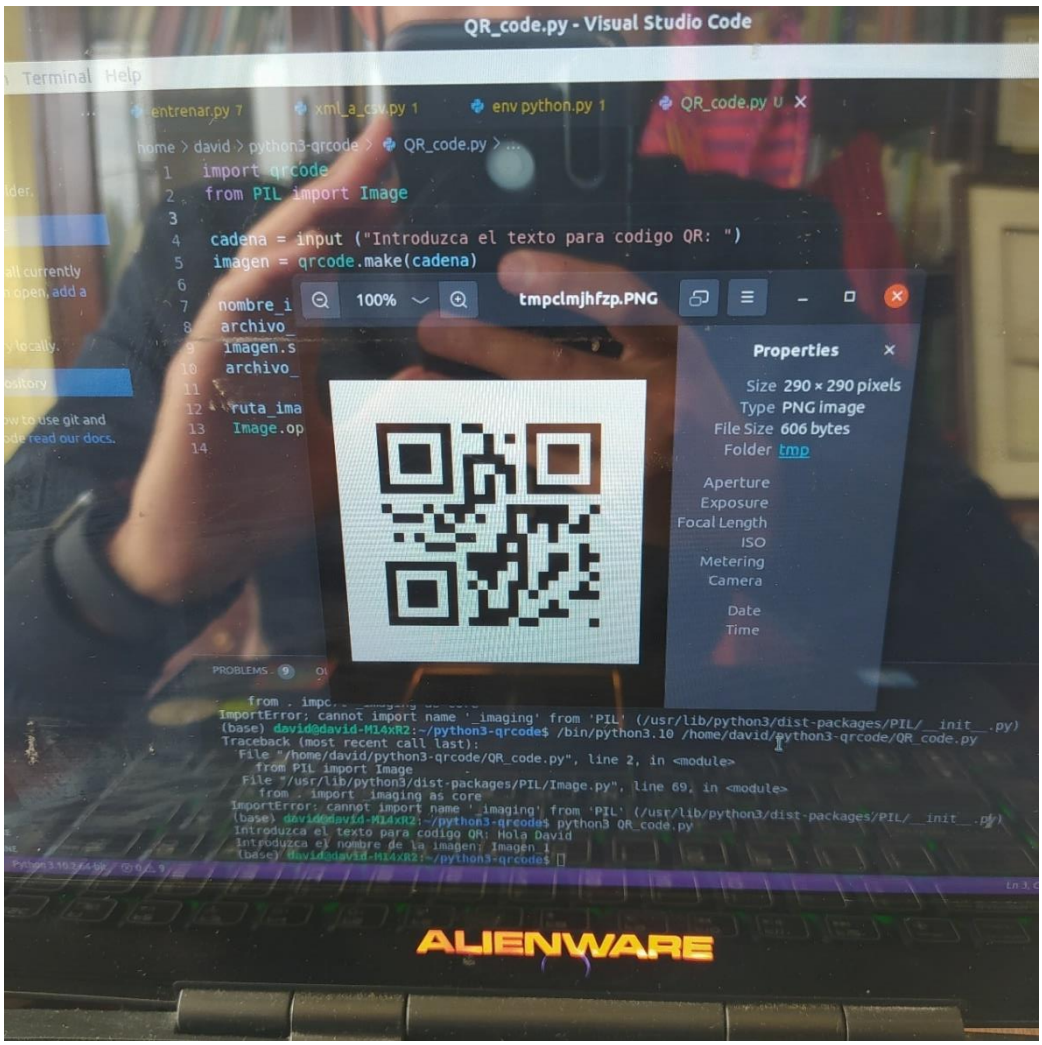

Fig. 41: Generación de códigos QR

Elaborado por: David Chávez

4.8.4 Etapa escaneado del código QR

En la última etapa del sistema, es la verificación de la información almacenada en el código QR que se había generado en la etapa anterior, para lo cual en este trabajo de titulación se ha optado utilizar un teléfono celular que tenga la aplicación de escaner de códigos QR para visualizar su información en el mismo dispositivo móvil como se puede observar en la Fig. 42

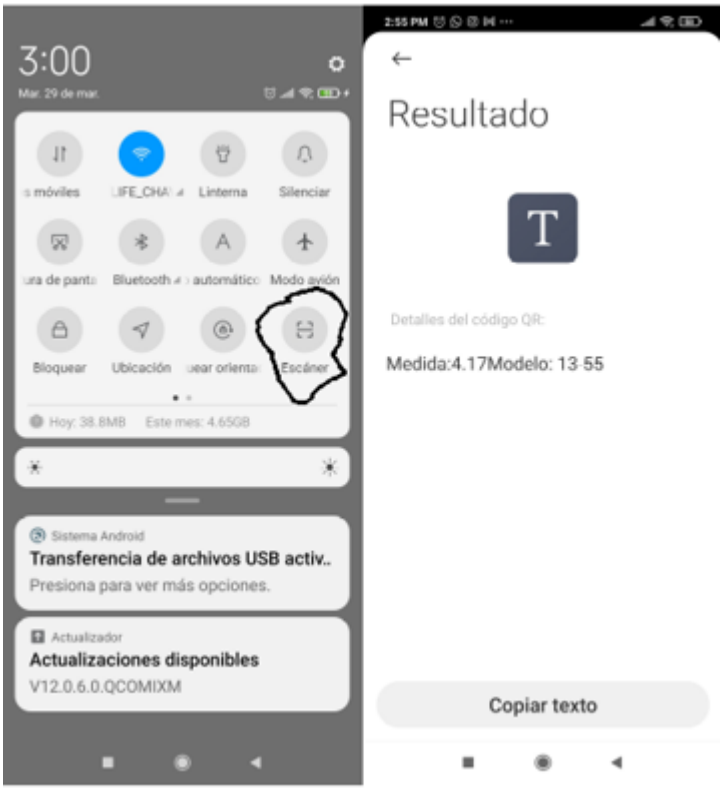

Fig. 42: Escaneer código QR desde teléfono móvil.

Elaborado por: David Chávez

4.9 Pruebas de funcionamiento e implementación del Sistema de Control

Después de verificar el funcionamiento de las diferentes etapas del sistema electrónico de control y etiquetado de molduras para cuadros en tiempo-real mediante machine learning, se dipone presentar las distintas pruebas de funcionamiento para validar el funcionamiento de cada etapa del proceso, por lo cual se inicia con la etapa de visión artificial mediante detección de objetos.

Lo primero es tener el dispositivo que se va a encargar de pre-entrenar, entrenar, procesar y predecir el tipo de moldura que esté presente frente a la cámara raspberry pi, en este caso se va utilizar una mini-computadora llamada Jetson Nano de NVIDIA como se observa en la Fig. 43

Fig. 43: Jeson Nano NVIDIA armada
Elaborado por: David Chávez

Como siguiente paso es muy importante instalar el sistema operativo dentro de una memory stick de 64 Gb que en este caso se instalará Ubuntu con sus respectivas librerias, la ventaja de utilizar la tarjeta Jetson Nano es que ya viene instalado el paquete de CUDA y API para Deep Learning, Vision por computadora, acelerador de computo y multimedia. Al momento de la instalación se puede instalar los diferentes modelos de detección de objetos, con los paquetes y librerías completas se puede empezar a entrenar el modelo como se muestra en la Fig. 44, aquí se realizó el cáculo del entrenamiento por época y se llegó a la conclusión de entrenamiento igual a 4 minutos y 26 segundos en la época 25.

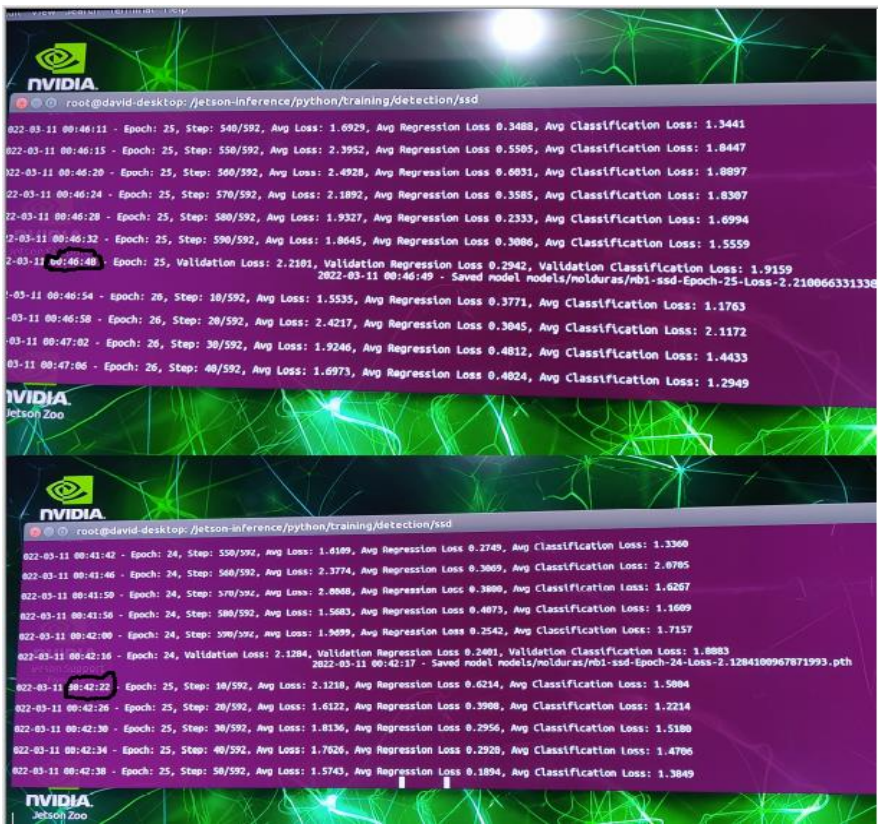

Fig. 44: Entrenamiento modelo molduras
Elaborado por: David Chávez

Después de entrenar el modelo de molduras que en este caso se la hizo para tres tipos llamados 12-1039, 13-55 y 45-14, el siguiente paso es cambiarlo al formato ONNX como se va a observar en la Fig. 45

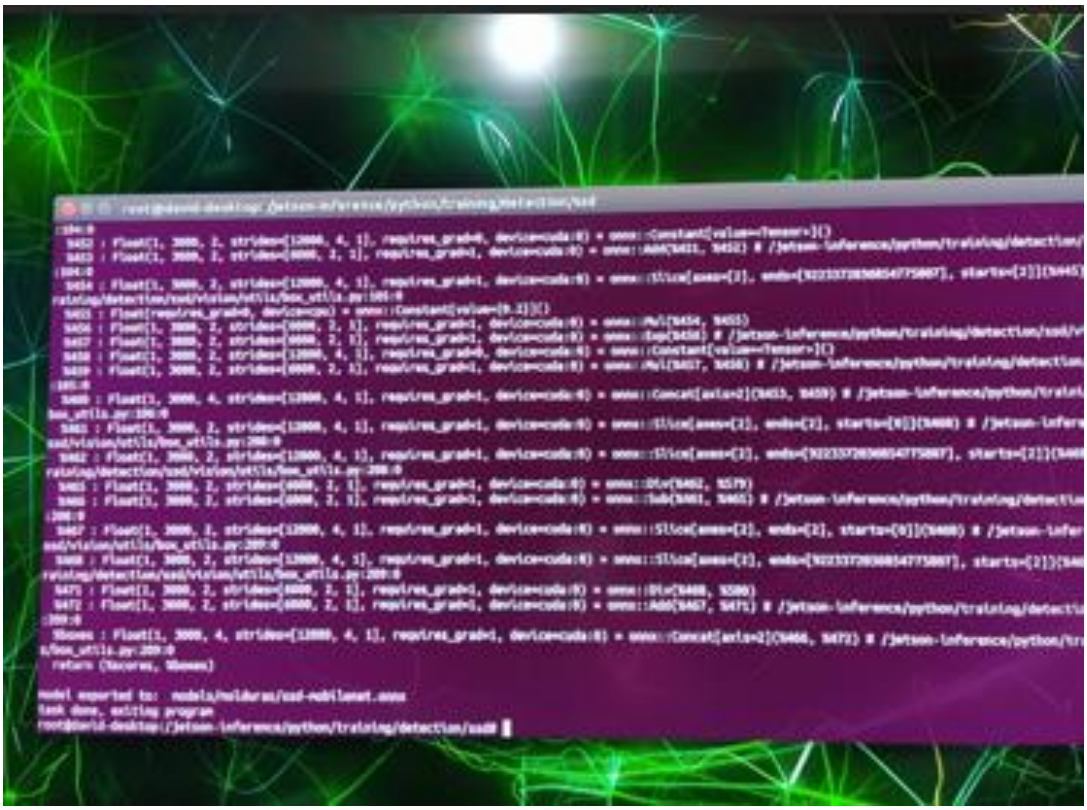

Fig. 45: Modelo molduras en formato ONNX

Elaborado por: David Chávez

Una vez generado el modelo entrenado, la siguiente etapa es de predicción que en este caso se realizó para tres tipos de molduras llamados: “13-55”, “12-1039” y “45-14” lo cual se observa en la Fig. 46 y el porcentaje de confianza en la Fig. 47 que es igual a 85,7% para el modelo 12-1039, 71,2% para el modelo 45-14 y 57,3% para el modelo 13-55 desde este ángulo de observación de la cámara raspberry pi. Aquí se observa que si tiene un alto porcentaje de confianza para reconocer el modelo 12-1039.

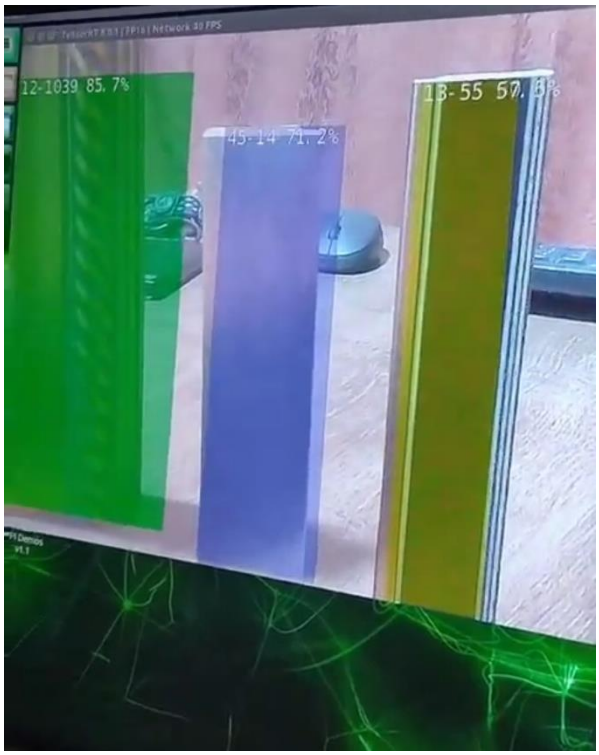

Fig. 46: Predicción tipo de moldura

Elaborado por: David Chávez

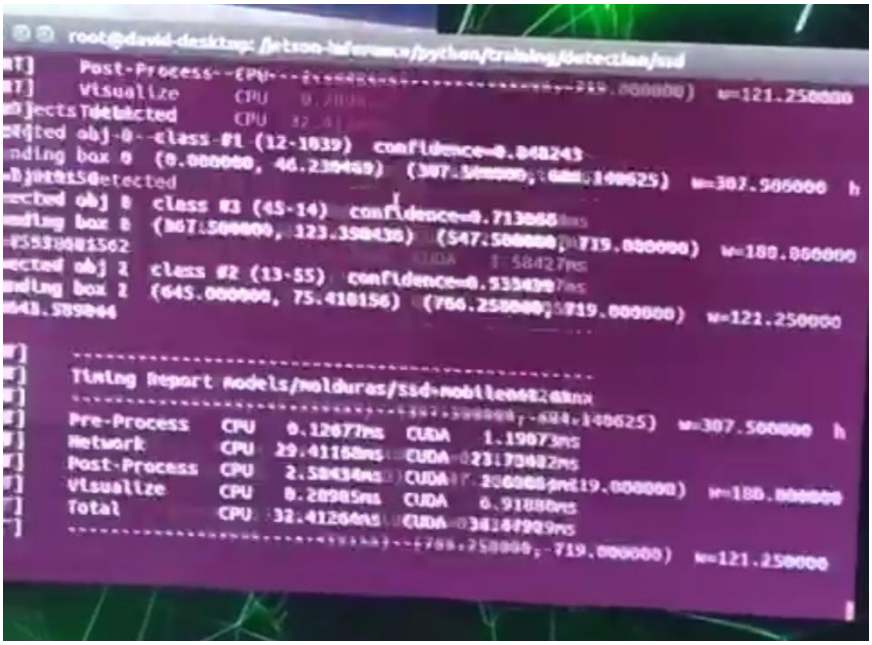

Fig. 47: Detección de la clase y confianza del modelo de moldura

Elaborado por: David Chávez

Después de la detección del tipo de moldura, implicaría la medición de la longitud de la tira de moldura en metros como se muestra en la Fig. 48 desde el dispositivo electrónico Arduino con su sensor ultrasónico HC-SR04 y mediante comunicación serial se transmite los datos desde Arduino IDE hacia la plataforma de Python que en este caso utilizaremos PyCharm.

Fig. 48: Adquisición de la longitud de la tira mediante Arduino Mega 2560

Elaborado por: David Chávez

El código de programación para calcular la distancia de la moldura está dada en el diagrama de flujo que se visualiza en la Fig. 49 y Fig.50

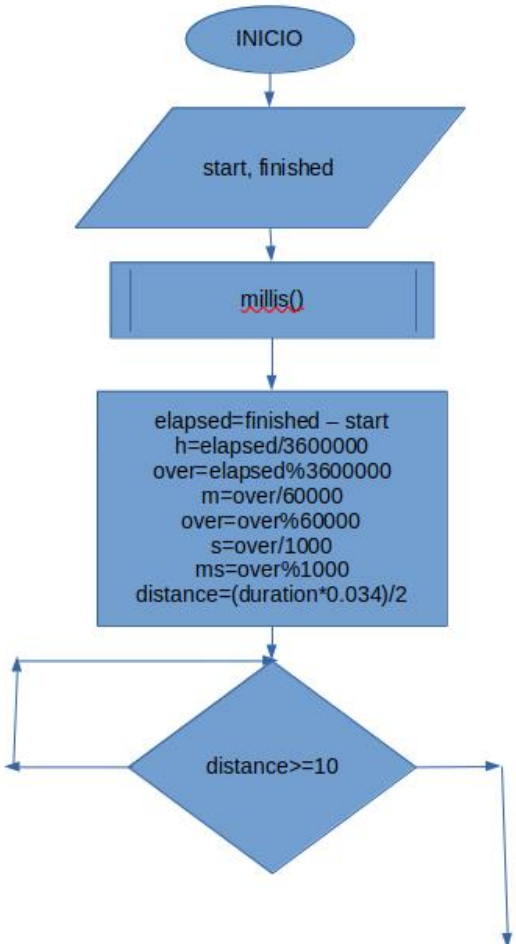

Fig. 49: Diagrama de flujo, adquisición medida de moldura

Elaborado por: David Chávez

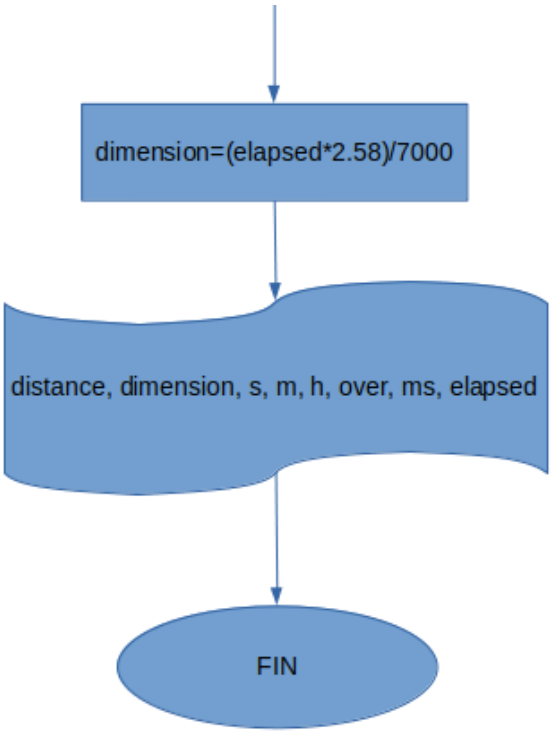

Fig. 50: Diagrama de flujo, adquisisción medida de moldura

Elaborado por: David Chávez

Finalmente, obtenido la detección del modelo de moldura mediante la mini-computadora Jetson Nano de NVIDIA a través de la cámara Raspberry Pi y la adquisición de datos de la medida de la tira de molduras, esta información será almacenada en un código QR para visualizarlo mediante un lector o escaner que en este caso se lo ha hecho desde un smartphone. A continuación, se puede observar en la Fig. 51, Fig. 52 y Fig.53

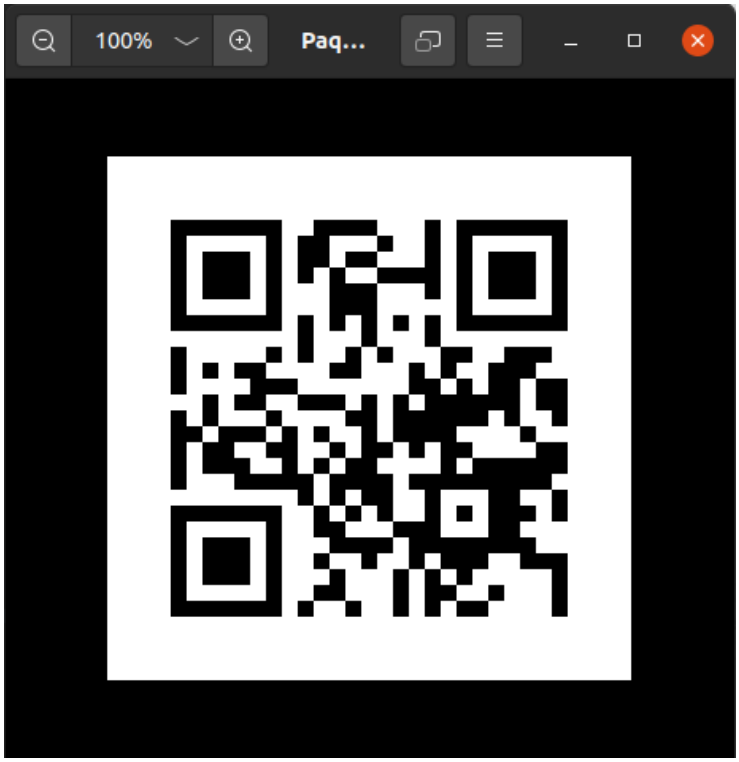

Fig. 51: Código QR generado

Elaborado por: David Chávez

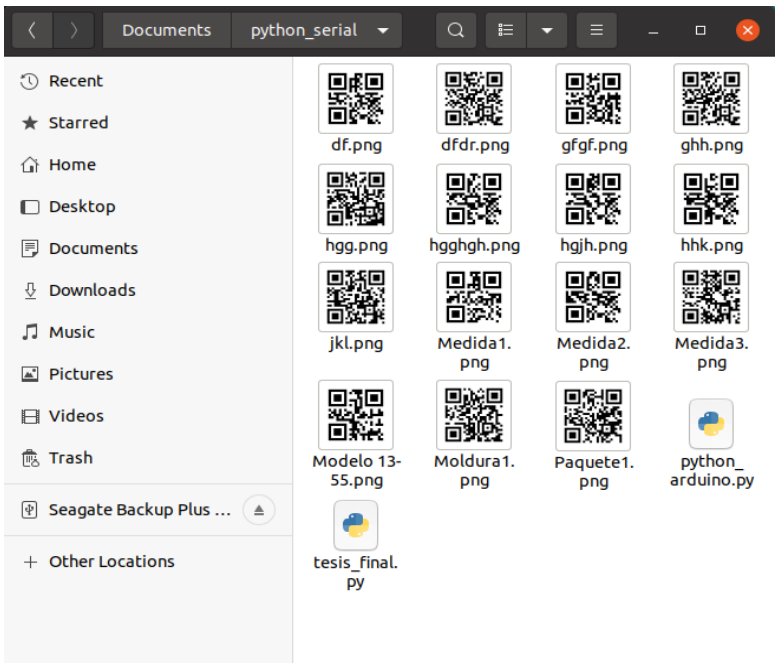

Fig. 52: Carpeta que almacena todos los códigos QR generados

Elaborado por: David Chávez

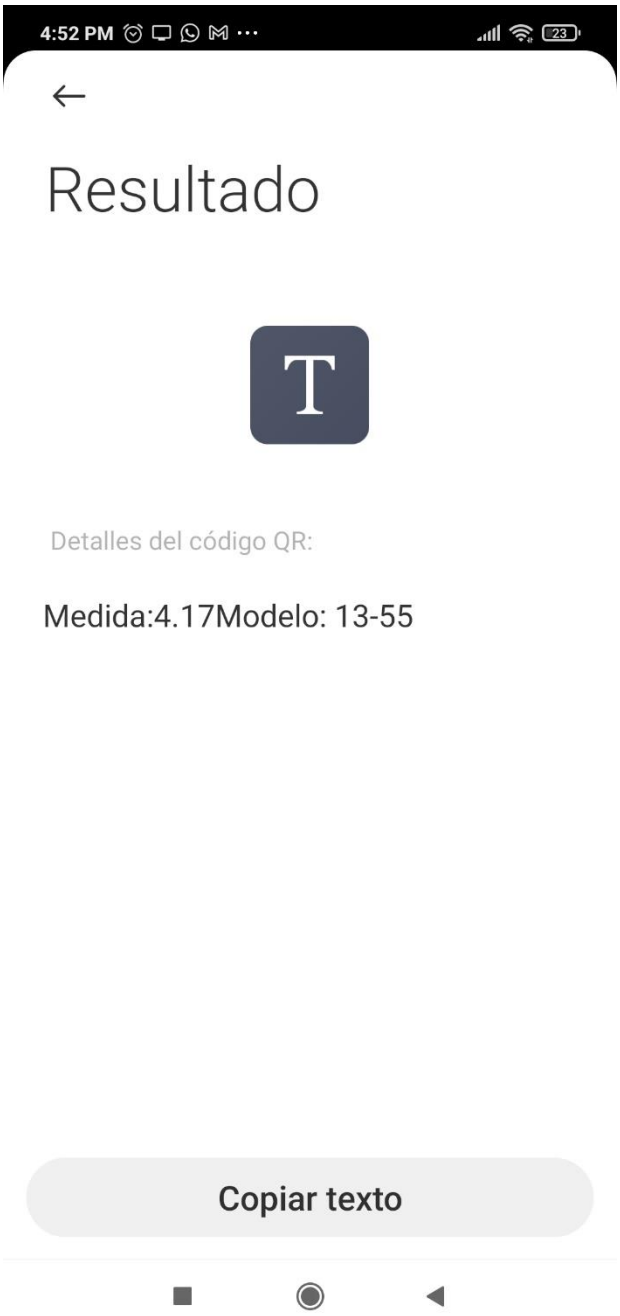

Fig. 53: Información almacenada y visualizada desde un scaner de códigos QR

Elaborado por: David Chávez

La captura de imágenes de los modelos de molduras “12-1039”, “13-55” y “45-14” se recopiló 2100 imagenes desde varios ángulos, fondos e iluminación para lo cual se puede observar en la Fig. 54, una parte del conjunto de datos adquiridos desde la herramienta Capture-camera del dispositivo Jetson nano a continuación.

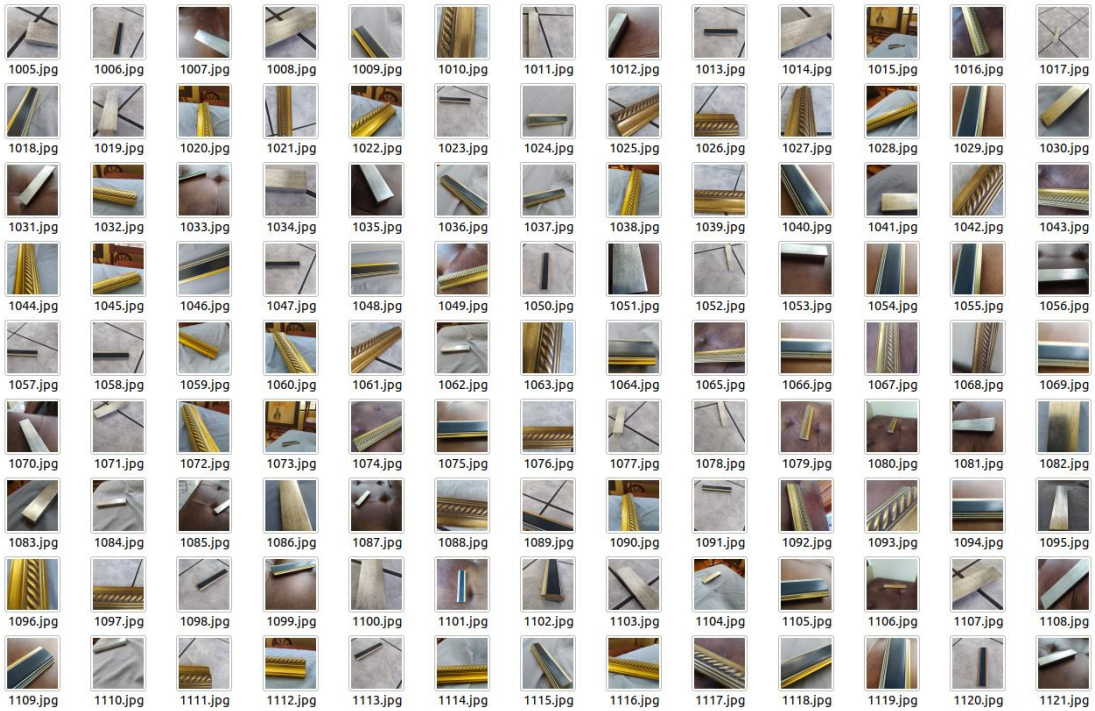

Fig. 54: Conjunto de datos de las molduras "12-1039", "13-55" y "45-14"

Elaborado por: David Chávez

El sistema electrónico de control y etiquetado de molduras para cuadros en tiempo-real mediante machine learning ha quedado implementado de la siguiente manera, como podemos observar en la Fig. 55 donde se encuentra la máquina encargada de transportar la tira de moldura mediante una banda tranportadora , en la Fig. 56 se visualiza la ubicación del sensor ultrasónico el cual está encargado de calcular el tamaño de la tira de moldura y la Fig.57 es la que está encargada de detectar cual es el modelo de la moldura.

Fig. 55: Máquina transportadora de moldura

Elaborado por: David Chávez

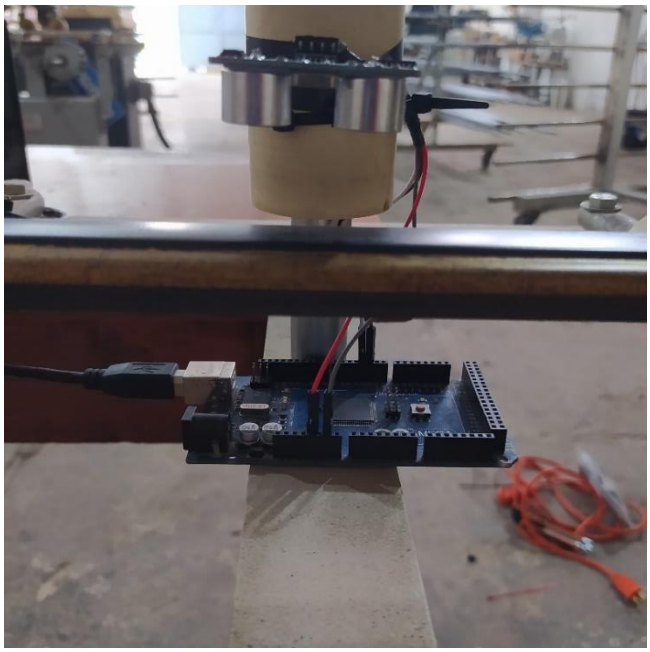

Fig. 56: Arduino Mega2560 y sensor ultrasónico HC-SR04, etapa medición moldura

Elaborado por: David Chávez

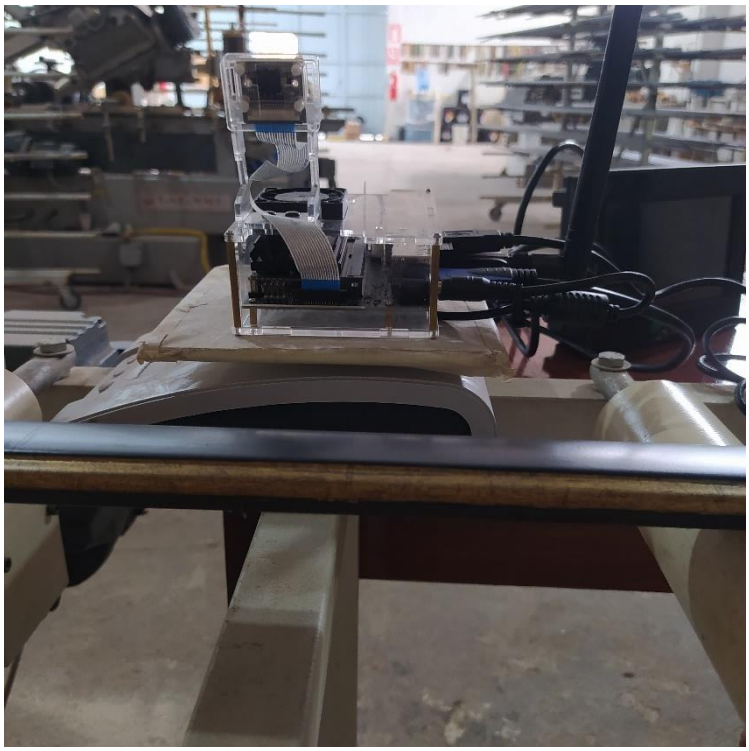

Fig. 57: Jetson Nano NVIDIA, etapa detección modelo de moldura

Elaborado por: David Chávez

CAPÍTULO V

CONCLUSIONES, RECOMENDACIONES, BIBLIOGRAFÍA Y ANEXOS

5.1 Conclusiones

- A través de la recopilación de información como en diferentes fuentes bibliográficas como es en artículos científicos, libros y páginas académicas de internet se logró diseñar e implementar la detección de tres tipos de molduras mediante aprendizaje supervisado lo cual pertenece al machine learning en tiempo-real.
- El tipo de sistema que se ha implementado viene dado por diferentes dispositivos electrónicos que han sido de gran utilidad y prácticos para poder cumplir con la arquitectura de reconocimiento de molduras y medición de la longitud de los diferentes tipos de moldura.
- Se ha realizado las diferentes pruebas de funcionamiento del sistema y se ha llegado a la conclusión de que cada una de las etapas del sistema electrónico de control y etiquetado de molduras para cuadros en tiempo-real mediante machine learning se ha obtenido el valor de confianza de la siguiente manera, 85,7% para el modelo 12-1039, 71,2% para el modelo 45-14 y 57,3% para el modelo 13-55 con tiempo aproximado de 4 minutos por época.
- Al momento de realizar los tiempos de entrenamiento de los diferentes tipos de moldura se ha tomado el tiempo de cada época para ver la utilidad del dispositivo Jetson Nano de NVIDIA por el uso de la GPU para ahorro de tiempo de entrenamiento y se ha llegado a la conclusión para 30 épocas con un total de 2100 imágenes, su duración a sido de 1 hora y 6 minutos.

5.2 Recomendaciones

- Es muy importante que los datos recopilados sean desde diferentes ángulos, puntos de vista de la cámara, orientaciones de objetos, condiciones de iluminación, diferentes fondos con el fin de crear un modelo resistente al ruido y muchos cambios en el entorno.
- En el caso de que el modelo de verificación de los distintos tipos de moldura no esta funcionando bien es recomendable aumentar 100 imágenes por clase para ir mejorando el porcentaje de confianza del sistema y reducir el error cuadrático para predecir el modelo de moldura.

- El sistema electrónico al momento de implementarlo es muy importante ubicarlo en un lugar estratégico para evitar que se mueva o moleste al operador de la máquina cuando esté realizando su trabajo.
- Se recomienda utilizar una cámara con más resolución para mejorar la captura de las fotos antes de empezar con el entrenamiento y procesamiento de imágenes en la Jetson Nano de NVIDIA.

BIBLIOGRAFÍA

[1]Nvidia developer (2022), [Online]. Disponible: https://developer.nvidia.com/deep-learning

[2] R. Flórez, J. M. Fernández, *Las redes neuronales artificiales,* España. Netbiblo. 2008

[3] R. Flórez, J. M. Fernández, *Las redes neuronales artificiales,* España. Netbiblo. 2008

[4]A. Bochkovskiy, Ch. Wang, & M. L. Hong-Yuan. "YOLOv4: Optimal Speed and Accuracy of Object Detection", *arXiv, vol1, Abril 2020*

[5] UDEMY (2022), [Online]. Disponible: https://www.udemy.com/course/deep-learning-con-keras/learn/lecture/22655249#content

[6] E. Ohbuchi, H. Hanaizumi, L. A. Hock, "Barcode Readers using the Camera Device in Mobile Phones", *IEEE 2004 International Conference on Cyberworlds*, pp. 260-265, Noviembre 2004

[7] Colaboratory (2022), [Online]. Disponible: https://colab.research.google.com/

[8]JetsonHacks(2014-2022).[Online].Disponible:https://www.jetsonhacks.com/2019/06/07/jetson-nano-gpio/

[9] I-Nigma (2008), [Online]. Disponible: http://www.i-nigma.com/

[10] D. H. Shin, J. Jung, & B. H. Chang, "The psychology behind QR codes: User experience perspective". *Computers in Human Behavior*, vol.*28*, p.p. 1417-1426, Marzo 2012.

[11] T. Falas, H. Kashani, "Two-Dimensional Bar-Code Decoding with Camera-Equipped Mobile Phones", *Fifth Annual IEEE International Conference Pervasive Computing and Communications Workshops (PerCom)*, pp. 597 – 600, 2007.

[12] D. Reilly, G. Smolyn and H. Chen, "Toward fluid, mobile, and ubiquitous interaction with paper usingrecursive 2D barcodes." *Pervasive Mobile InteractionDevices 2007 (PerMID 2007), workshop at Pervasive 2007*, Toronto, Canada, 2007.

[13] N. Fujimura, M. Doi, "Collecting students' degree of comprehension with mobile phones", *Proceedings of the 34th annual ACM SIGUCCS conference on User servicestable of contents*, Edmonton, Alberta, Canada pp. 123 – 127, ISBN:1-59593-438-3, Canada 2006.

[14] V. Sati, S. Sánchez, M., N. Shoeibi, A. Arora, & J. M. Corchado, "Face Detection and Recognition, Face Emotion Recognition Through NVIDIA Jetson Nano.", In *International Symposium on Ambient Intelligence* (pp. 177-185). Springer, Cham. (2020, June).

[15]A. Roman (2021), [Online]. Disponible:https://proyectoarduino.com/arduino-mega-2560/

[16]Sandorobotics(2019),[Online].Disponible:https://sandorobotics.com/producto/af-3099/

[17]Hetpro(2021),[Online].Disponible:https://hetpro-store.com/TUTORIALES/sensor-hc-sr04/

[18] R. S. Olson, Python Machine Learning, Pennsylvania, University of Pennsylvania, 2015

[19]Jetbrains(2022),[Online].Disponible:https://www.jetbrains.com/es-es/lp/pycharm-anaconda/

[20]Arduino (2022), [Online]. Disponible: https://www.arduino.cc/

[21] NVIDIA (2022), [Online]. Disponible: https://github.com/dusty-nv/jetson-inference/blob/master/docs/pytorch-collect.md

[22] A. Younis, L. Shixin, S. JN, Z Hai, "Real-Time Object Using Pre-Trained Deep Learning Models MobileNet-SSD", In Proceedings of 2020 the 6th International Conference on Computing and Data Engineering, (pp. 44-48), 2020.

ANEXOS

Anexo 1.

Instalación sistema operativo en la tarjeta Jetson Nano NVIDIA

Lo primero para poder instalar el sistema operativo en la tarjeta Jetson Nano NVIDIA, es descargar e instalar el JetPack que a continuación se explica:

1. Descargar un software llamado Balena Etcher, el cual te permite flashear el sistema operativo en la tarjeta SD memory stick de 64GB para que se convierta en un disco duro como se tiene en las computadoras y cuando se encienda la Jetson Nano sepa que de ahi debe tomar el sistema operativo, como se visualiza en la Fig. 58.

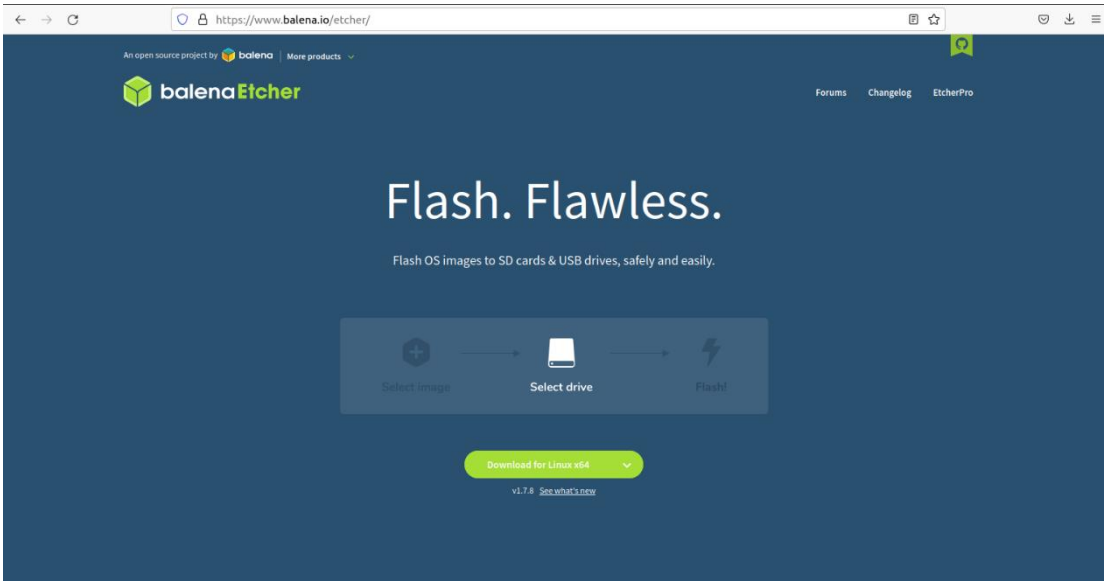

Fig. 58: Aplicación para flashear en memory SD 64GB
Elaborado por: David Chávez

2. Descargar la imagen .iso en este caso va ser Jetpack 4.5.1, ya viene con todos los drivers que se va necesitar, además viene con Ubuntu, en este caso la tarjeta es de 4GB y se toma la opción que esta a lado de una flecha roja como se observa en la Fig. 59.

Fig. 59: Instalar Jetpack

Elaborado por: David Chávez

3. Se carga el sistema operativo que se descargó anteriormente para la tarjeta Jetson Nano de 4GB como se observa en la Fig. 60.

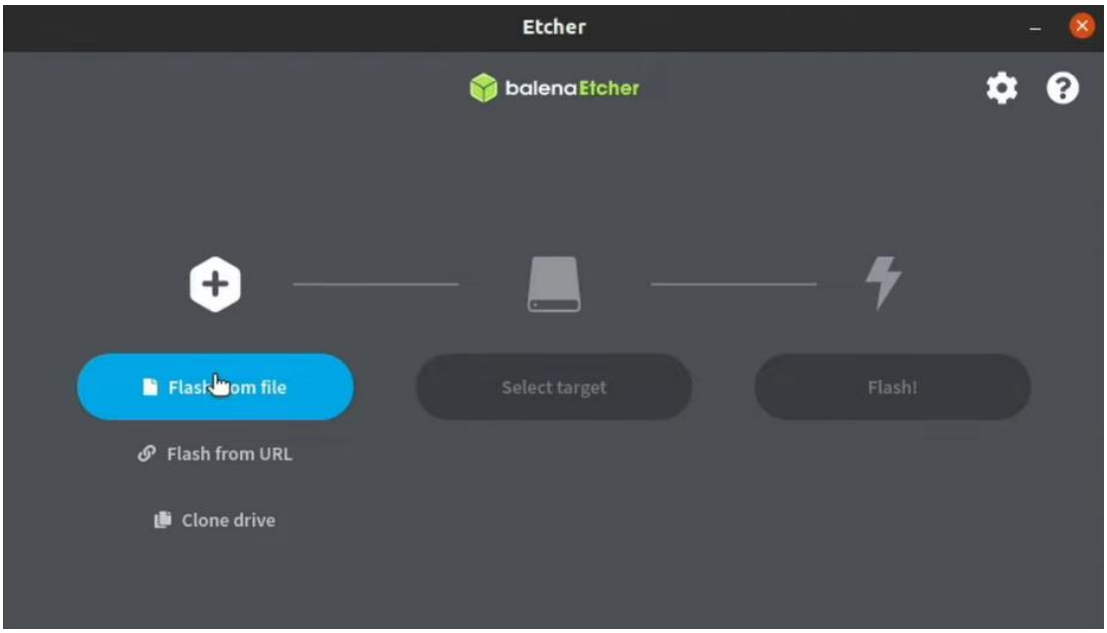

Fig. 60: Cargar sistema operativo en tarjeta SD de 64GB

Elaborado por: David Chávez

4. Luego que se haya elegido la imagen del sistema operativo, se elige donde va ser instalado y ente caso se tomaŕa la tarjeta SD de 64GB de la siguiente manera como se visualiza en la Fig. 61 y Fig.62.

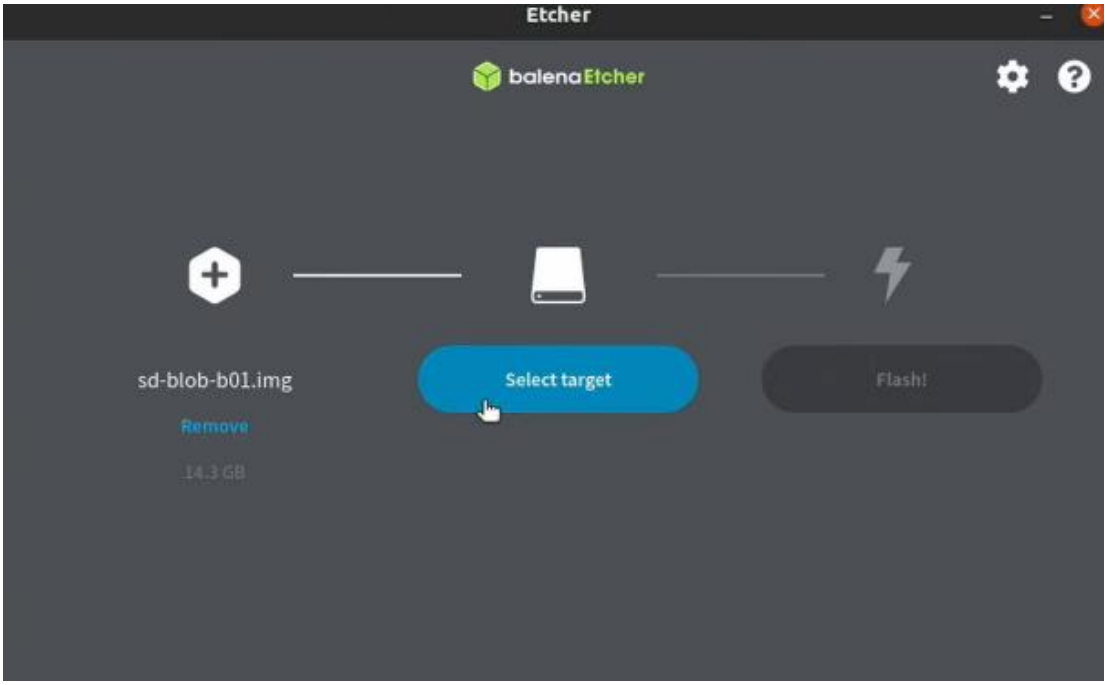

Fig. 61: Elección tarjeta SD 64GB para instalar sistema operativo parte 1
Elaborado por: David Chávez

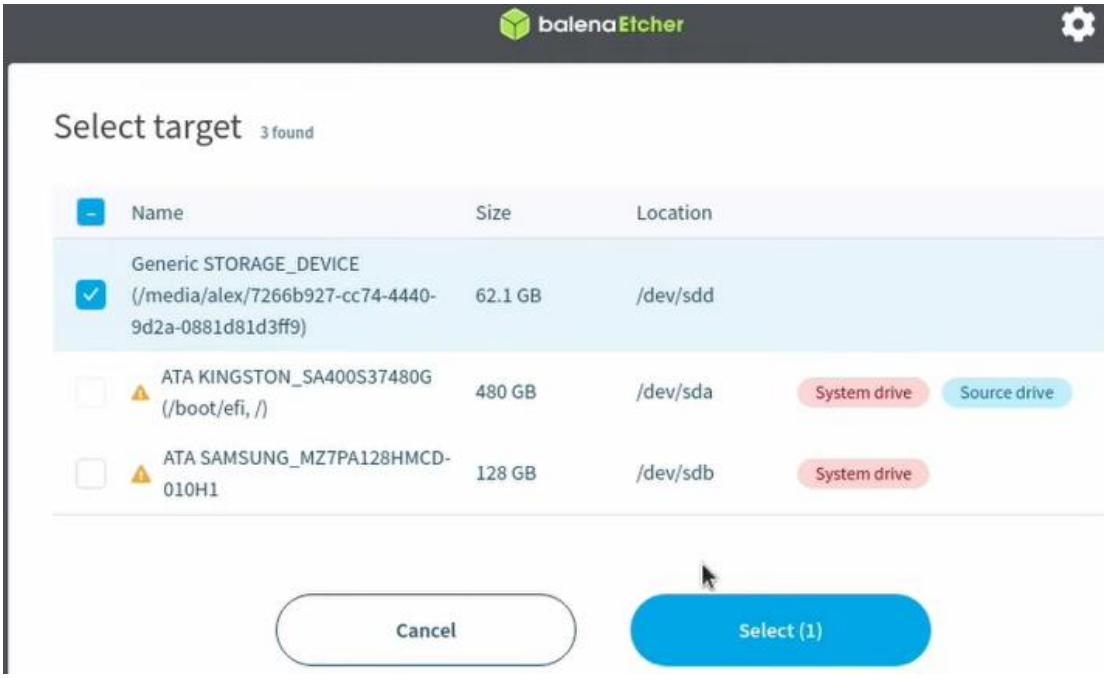

Fig. 62: Elección tarjeta SD 64GB para instalar sistema operativo parte 2
Elaborado por: David Chávez

5. Finalmente, damos clic en Flash para que empiece a flashear el sistema operativo en la tarjeta SD de 64 GB, tomará 15 minutos en terminar el proceso como se puede observar en la Fig. 63, Fig. 64 y Fig. 65.

Fig. 63: Flasheo memory SD 64GB parte 1

Elaborado por: David Chávez

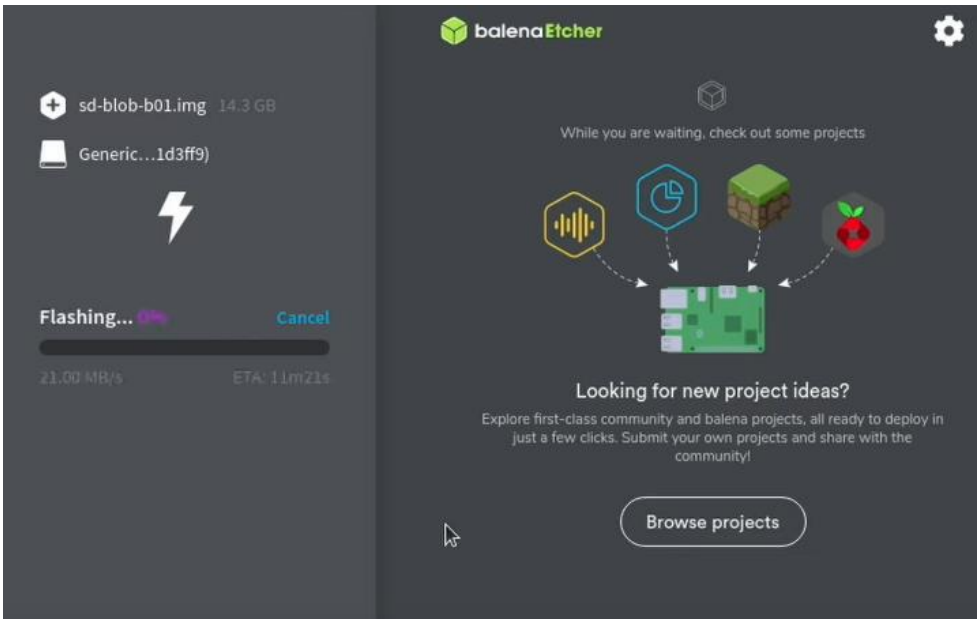

Fig. 64: Flasheo memory SD 64GB parte 2

Elaborado por: David Chávez

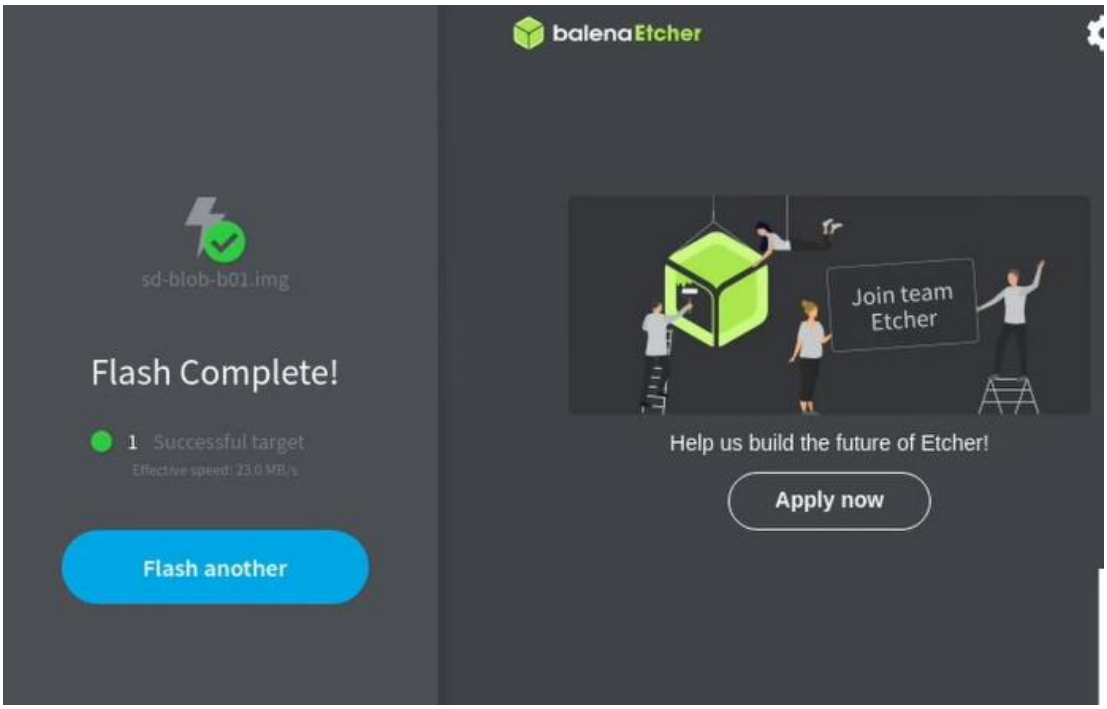

Fig. 65: Flasheo memory SD 64GB parte 3

Elaborado por: David Chávez

Anexo 2.

Configuración de la tarjeta Jetson Nano NVIDIA

1. Colocar la tarjeta SD de 64GB dentro de la tarjeta Jetson Nano NVIDIA y encender, en ese momento aparecerá en la pantalla la Fig. 66 y aceptar los téminos de la licencia.

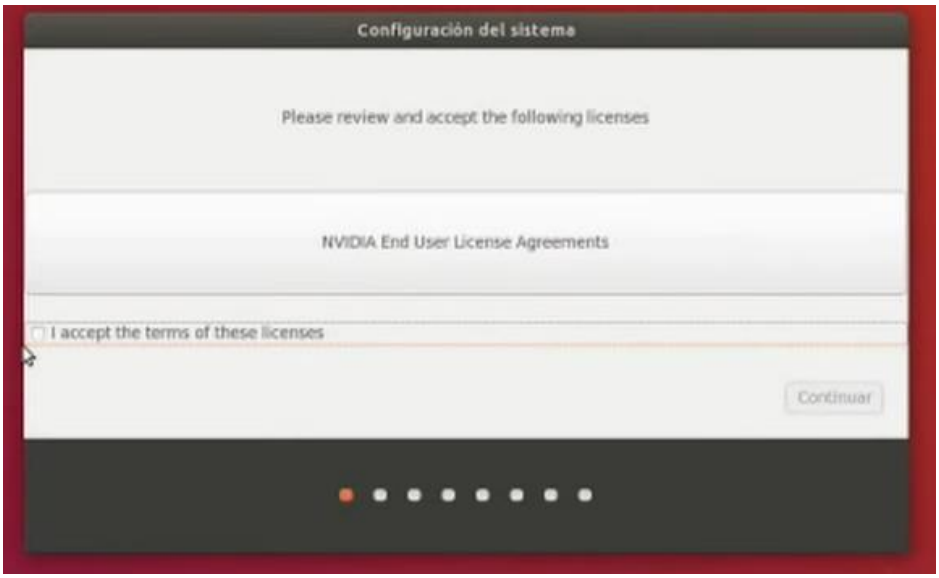

Fig. 66: Configuración del sistema operativo

Elaborado por: David Chávez

2. Elegir el idioma, en este caso español como se visualiza en la Fig. 67.

Fig. 67: Configuración del sistema operativo

Elaborado por: David Chávez

3. Elegir el tipo de red inalámbrica como se observa en la Fig. 68, que se desee conectar en el caso que se tenga la tarjeta inalámbrica conectada a la tarjeta Jetson Nano o también existe la posibilidad de conectarse de manera alámbrica mediante cable Ethernet al modem.

Fig. 68: Conexión inalámbrica a la red de Internet

Elaborado por: David Chávez

4. Elegir el tipo de fuente de alimentación que esta conectada la tarjeta Jetson Nano NVIDIA, en este caso se ha utilizado una fuente de alimentación que va conectado al conector de barril para tener mejores prestaciones de hardware como se observa en la Fig. 69 o la otra opción sería conectase mediante una micro-usb pero las prestaciones de hardware se reducen a la mitad como se observa en la Fig. 70.

Fig. 69: Alimentación tarjeta Jetson Nano NVIDIA opción 1

Elaborado por: David Chávez

Fig. 70: Alimentación tarjeta Jetson Nano NVIDIA opción 2

Elaborado por: David Chávez

Se da continuar y va empezar hacer toda la configuración del sistema como se observa en la Fig. 71

Fig. 71: Inicia configuración del sistema

Elaborado por: David Chávez

5. Finalmente, cuando se haya terminado de configurar el sistema operativo, se va abrir el terminal vamos a digitar dos códigos para actualizar los paquetes del sistema operativo como se se observa en la Fig. 72 y Fig. 73

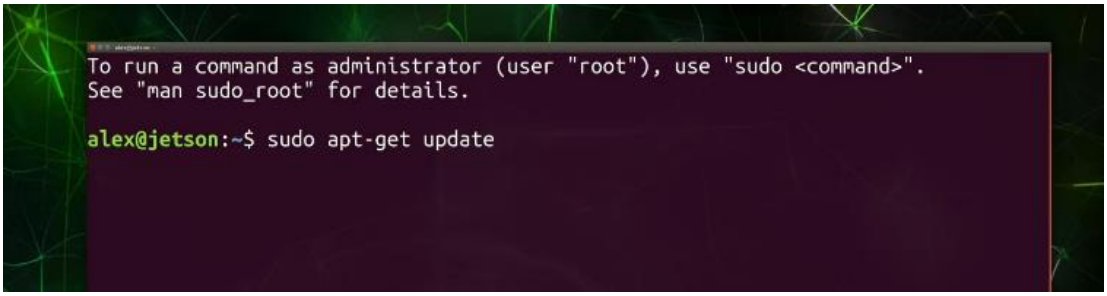

Fig. 72: Actualizar paquetes en sistema operativo parte 1

Elaborado por: David Chávez

Fig. 73: Actualizar paquetes en sistema operativo parte 2

Elaborado por: David Chávez

Anexo 3.

Instalar repositorio para Detección de objetos en la Tarjeta Jetson Nano NVIDIA

1. Abrir el terminal para crear una carpeta llamada Deteccion-objetos con el comando:

$ mkdir Deteccion-objetos

En esta sección se ingresa a la carpeta Deteccion-objetos con el siguiente comando:

$ cd Detección-objetos

Cuando se esta dentro de la carpeta, se clona el repositorio que este caso será el siguiente como observamos en la Fig. 74

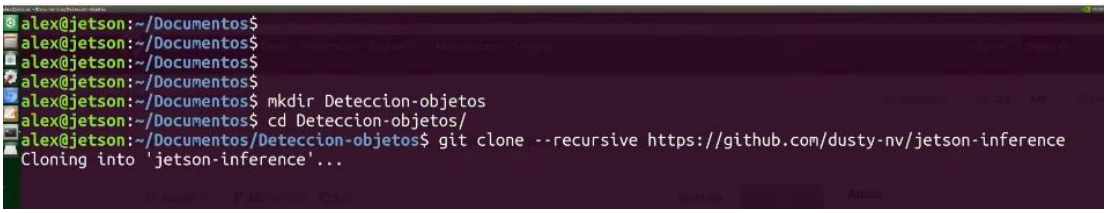

Fig. 74: Clonar repositorio Deteccion-objetos

Elaborado por: David Chávez

2. Una vez que se descargó el repositorio, se puede navegar en la carpeta jetson inference ingresando con el siguiente comando:

$ cd jetson-inference

Cuando esta en la carpeta jetson-inference se puede visualizar todo lo que tiene esta carpeta mediante el comando:

$ ls -l

Aquí va existir varias carpetas y varios archivos como se puede observar en la Fig. 75

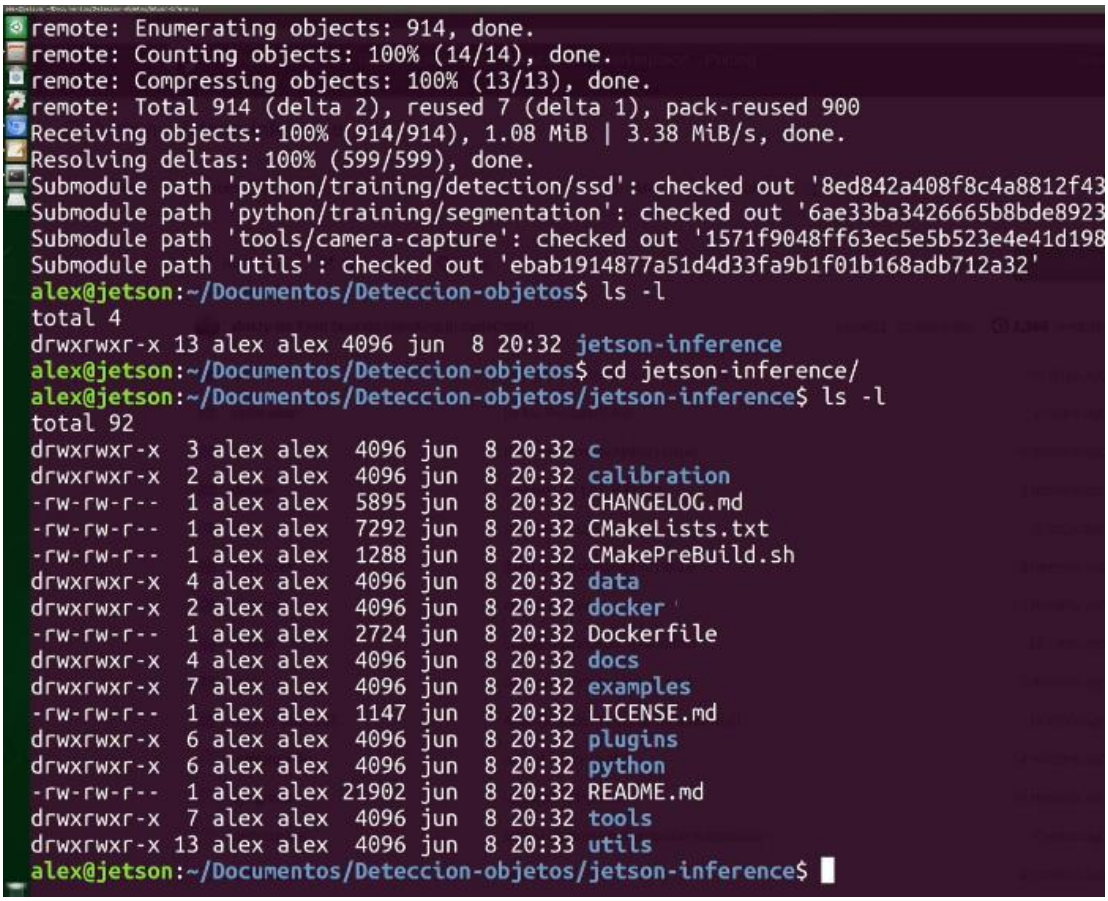

Fig. 75: Carpeta jetson-inference

Elaborado por: David Chávez

3. Se va correr el docker el cual esta ubicado en la carpeta de jetson-inference con el siguiente comando:

$ docker/run.sh

De esta manera ingresamos como super usuario al docker, esto va hacer que enves de manejarlo desde la tarjeta Jetson nano NVIDIA, se lo va utilizar desde una máquina virtual más pequeña dentro de la Jetson nano como se observa en la Fig. 76

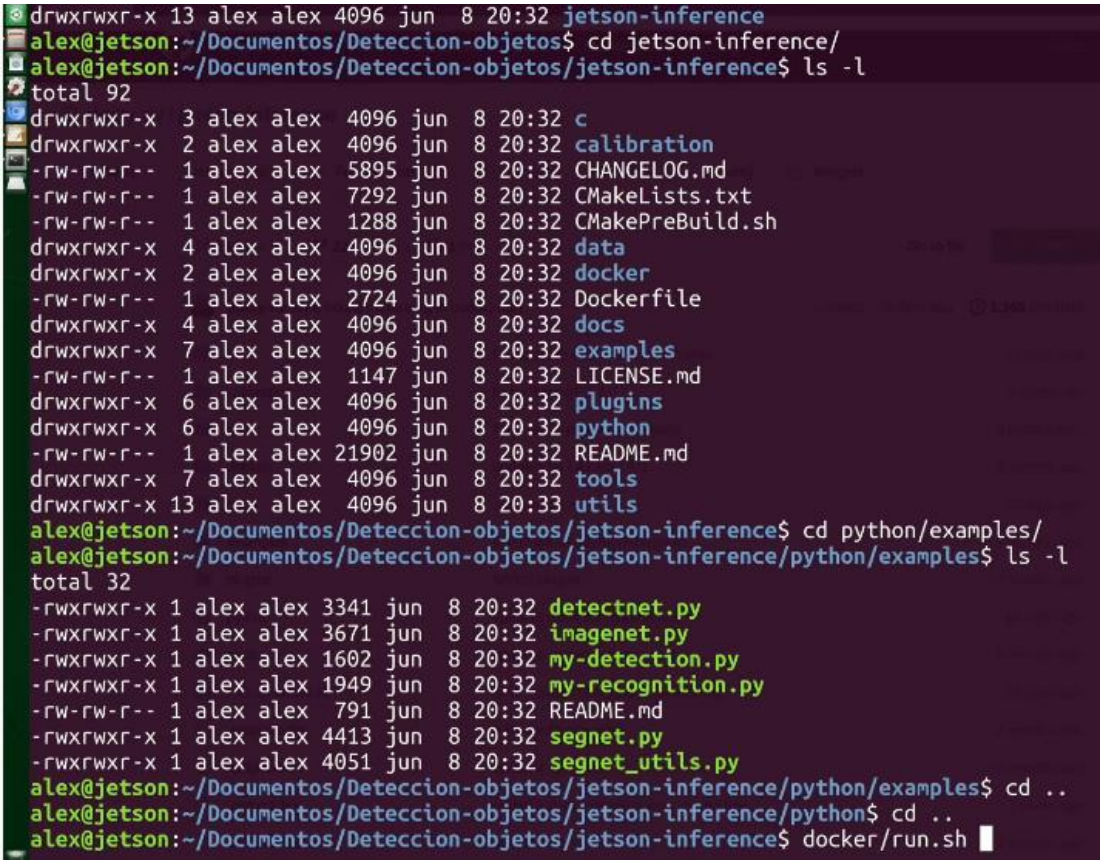

Fig. 76: Correr máquina virtual docker en la tarjeta Jetson nano NVIDIA

Elaborado por: David Chávez

4. Aparecerá la siguiente pantalla, en la cual se va elegir que tipo de modelos se va descargar para los entrenamientos de lo que se necesite, como se puede observar en la Fig. 77

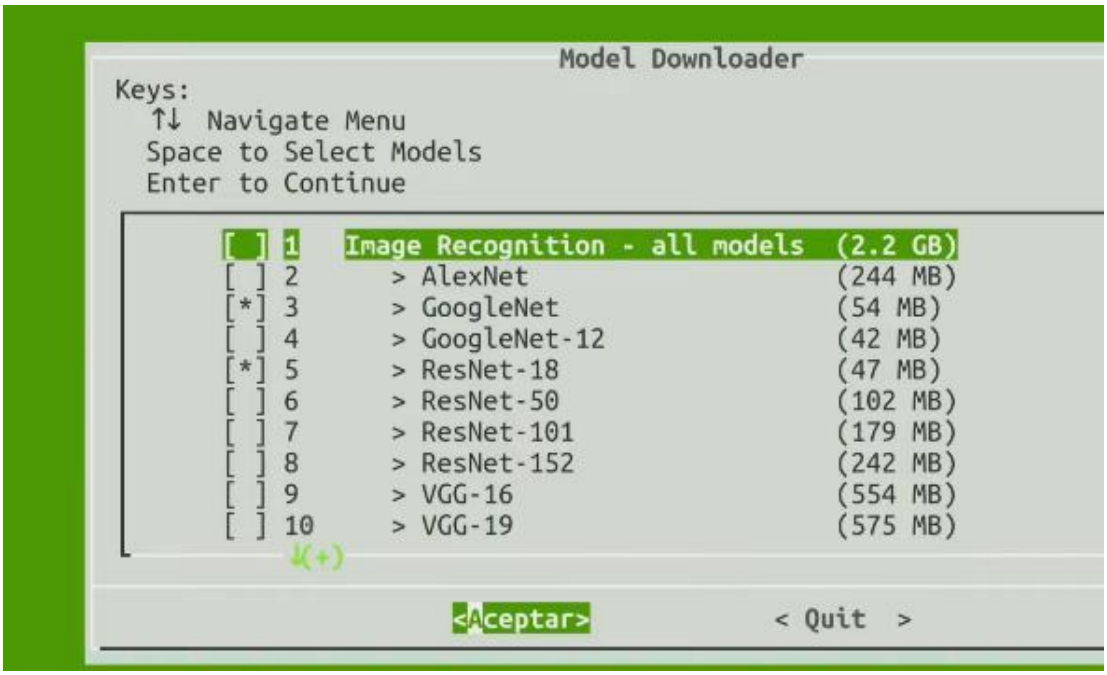

Fig. 77: Elegir modelos preentrenados

Elaborado por: David Chávez

5. Después de seleccionar los modelos de entrenamiento, se da aceptar e inmediatamente empieza a descargar lo que se seleccionó, como se visualiza en la Fig. 78

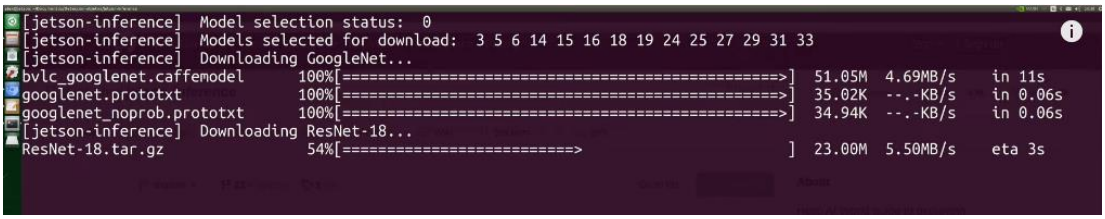

Fig. 78: Descarga de los modelos preentrenados

Elaborado por: David Chávez

Finalmente, para instalar Pytorch, debe colocar el siguiente comando:

$ cd jetson-inference/build

$ ./install-pytorch.sh

Inmediatamente aparecé la ventana, la cual es la Fig. 79, como es este caso, se eligió Pytrorch para python version 3.6.

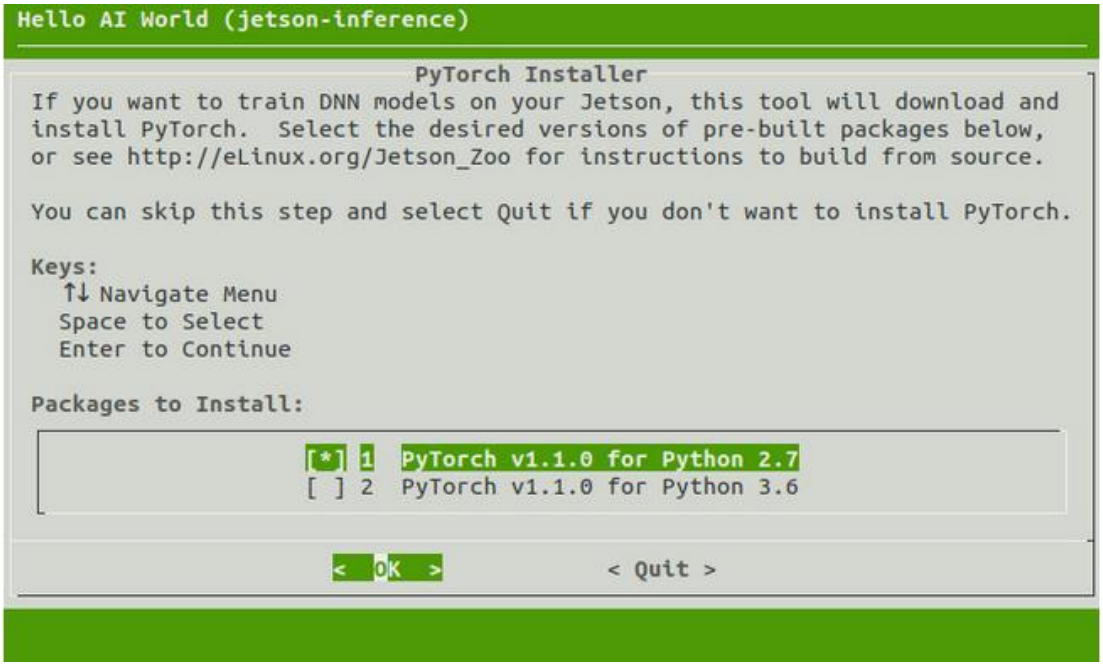

Fig. 79: Instalación Pytorch versión python 3.6

Elaborado por: David Chávez

Anexo 4.

Instalar GPIO en Jetson Nano

Para acceder a los pines GPIO en la tarjeta Jetson Nano se utiliza la biblioteca Jetson.GPIO, esta es una librería de la tarjeta Jetson. Esta biblioteca ha sido modificada de la biblioteca RPI GPIO (Rasperry). Se puede crear aplicaciones en python con Jetson.GPIO. Esta biblioteca se la puede descargar desde: https://github.com/NVIDIA/jetson-gpio.

Se puede instalar Jetson.GPIO en la tarjeta Jetson Nano de NVIDIA utilizando pip de Python. Para lo cual hay que abrir el Terminal y escribir el siguiente comando: [8]

```
$ sudo pip install Jetson.GPIO
```

Si se usa Python3, lo más seguro es que se va utilizar pip3 para instalar en la Jetson Nano NVIDIA.[8]

```
$ sudo pip3 install Jetson.GPIO
```

Otra parte muy importante es configurar los permisos de seguridad en la tarjeta jetson Nano de NVIDIA para compilar los programas. Crea un nuevo usuario de grupo gpio. Despúes, se agrega una cuenta dentro del grupo gpio con los siguientes comandos: [8]

```
$ sudo groupadd -f -r gpio
    $ sudo usermod -a -G gpio <your_account>
```

También es importante copiar el siguiente archivo 99-gpio.rules dentro de la carpeta /etc/udev/rules.d/ . El archivo 99-gpio.rules usualmente este habilitado en /usr/local/lib/python3.6/dist-packages/. Es posible verificar utilizando el siguiente comando para python 3, cambia python por Python2.7.x [8]

```
    $ python3 -m site
```

Se debe ver el parche de la carpeta dist-packages de Python. Por ejemplo se tiene el parche para la carpeta usr/local/lib/python3.6/dist-packages/ así se puede mover el archivo 99-gpio.rules dentro de la carpeta /etc/udev/rules.d/ con el siguiente comando. [8]

```
$ sudo cp /usr/local/lib/python3.6/dist-packages/Jetson/GPIO/99-gpio.rules /etc/udev/rules.d/
```

Después de que se complete la instalación, se puede verificar la instalación exitosa con el siguiente comando esto se presenta en la Fig. 16. [8]

```
$ python3
>>> import Jetson.GPIO as GPIO
    >>> GPIO.JETSON_INFO
>>> GPIO.VERSION
```

Anexo 5.

Instalar PyTorch

Instalación PyTorch

Al momento de de instalar PyTorch, se tiene dos opciones:

La primera es instalar PyTorch cuando creó el proyecto, por ende ya debería estar instalado en la Jetson Nano NVIDIA para usarlo.

La segunda opción es utilizando un contenedor de Docker y desea continuar con el aprendizaje de transferencia, puede instalarlo de la siguiente manera con los siguientes comandos desde la terminal:

$ cd jetson-inference/build
$ ./install-pytorch.sh

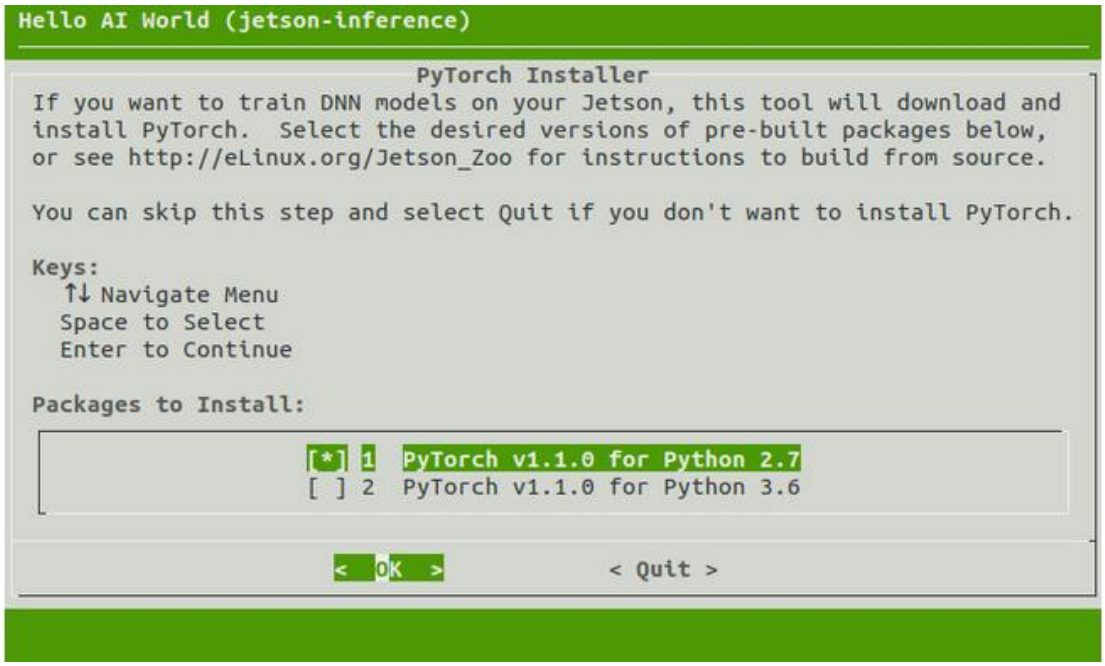

Fig. 80 Instalador PyTorch [21]

Lo cual se visualiza en la Fig. 80, para el presente caso se instaló PyTorch Para Python 3.6

Nota muy importante: Si se requiere que funcione de manera correcta PyTorch para el entrenamiento de detección de objetos es recomendable tener instalado JetPack 4.4 o la versión más actual.

Para comprobar que PyTorch este bien instalado y funcione correctamente tu GPU, se executa el siguiente comando en la terminal, además se utiliza Python 3.6 para correr con **python3** y si la versión es más baja se utilizará **python**. [21]

```
>>> import torch
>>> print(torch.__version__)
>>> print('CUDA available: ' + str(torch.cuda.is_available()))
>>> a = torch.cuda.FloatTensor(2).zero_()
>>> print('Tensor a = ' + str(a))
>>> b = torch.randn(2).cuda()
>>> print('Tensor b = ' + str(b))
>>> c = a + b
>>> print('Tensor c = ' + str(c))
```

Las dos siguiente lineas de código te permite conocer la versión de PyTorch está instalada

```
>>> import torchvision
>>> print(torchvision.__version__)
```

Anexo 6.

Montar Swap

Para el presente caso se va a usar la tarjeta Jetson Nano para lo cual se debe montar 4 GB de espacio de swap para utilizar la memoria en el entrenamiento, para lo cual dentro del contenedor en la tarjeta Jetson nano se va compilar en la terminal los siguientes comandos: [21]

```
sudo systemctl disable nvzramconfig
sudo fallocate -l 4G /mnt/4GB.swap
sudo mkswap /mnt/4GB.swap
sudo swapon /mnt/4GB.swap
```

Después se alojará el siguiente directorio ***etc/fstab*** en la tarjeta Jetson Nano para realizar un cambio en el archivo de texto con la última línea en este archivo.

```
/mnt/4GB.swap  none  swap  sw 0  0
```

Anexo 7.

Código de programación Arduino para medir la tira de moldura

dimension_moldura2

```
// Define Trig and Echo pin:
#define trigPin 2
#define echoPin 3
unsigned long start, finished, elapsed;
// Define variables:
long duration;
int distance;
float dimension;
int m=0;
//String miArray[10];

void setup() {
  // Define inputs and outputs:
  Serial.begin(9600);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(4, INPUT); // the start button
  pinMode(5, INPUT); // the stop button
  //Serial.println("Press 1 for Start/reset, 2 for elapsed time");

}

void displayResult()
{
 float h, m, s, ms;
 unsigned long over;
elapsed = finished - start;
 h = int(elapsed / 3600000);
 over = elapsed % 3600000;
 m = int(over / 60000);
 over = over % 60000;
```

Fig. 81: Codificación para medir tira de moldura parte 1

Elaborado por: David Chávez

dimension_moldura2

```
// Serial.println(elapsed);
// Serial.print("Elapsed time: ");
// Serial.print(h, 0);
// Serial.print("h ");
// Serial.print(m, 0);
// Serial.print("m ");
// Serial.print(s, 0);
// Serial.print("s ");
// Serial.print(ms, 0);
// Serial.println("ms");
// Serial.println();

}


void loop() {

  // Clear the trigPin by setting it LOW:
  digitalWrite(trigPin, LOW);
  delayMicroseconds(5);

  // Trigger the sensor by setting the trigPin high for 10 microseconds
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  // Read the echoPin, pulseIn() returns the duration (length of the pul
  duration = pulseIn(echoPin, HIGH);
  // Calculate the distance:
  distance = duration * 0.034 / 2;
```

Fig. 82: Codificación para medir tira de moldura parte 2

Elaborado por: David Chávez

dimension_moldura2

```
 //Serial.println(" m");
displayResult();
if (distance>10)
{
 start = millis();
 delay(200); // for debounce

 //Serial.println("Started...");

 //Serial.println("Dimension final:" + String(dimension, 2));
 }

}



else if (distance >= 10)
{
start = millis();
delay(200); // for debounce
//Serial.println("Started...");
m=0;
Serial.println("Medida:" + String(dimension, 2));

}



}
```

Fig. 83: Codificación para medir tira de moldura parte 3

Elaborado por: David Chávez

Anexo 8.

Manual de operación del sistema electrónico

1. Encender la tarjeta Jetson Nano de la empresa NVIDIA, esta debe estar conectada a la fuente de alimentación y muy importante, cuando la fuente de alimentación sea el conector de barril, se debe colocar un jumper para que empiece alimentar a la tarjeta Jetson Nano como se observa en la Fig 84

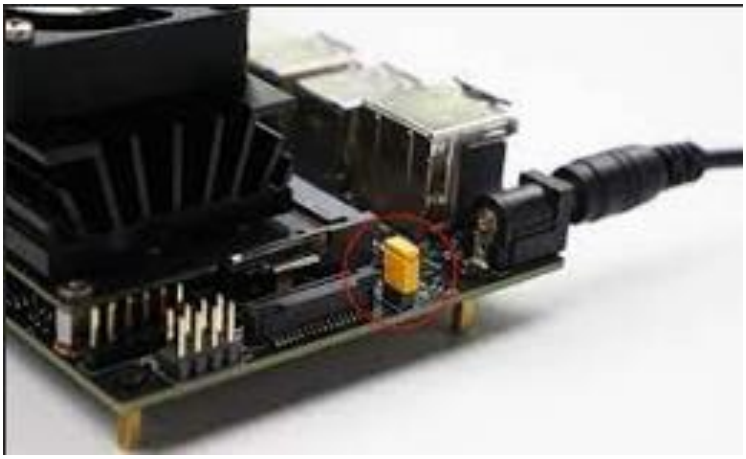

Fig. 84: Alimentación de energía a la tarjeta Jetson Nano

Elaborado por: David Chávez

2. Conectar el cable HDMI de va desde la pantalla que se esté utilizando en ese momento, inclusive puede ser una TV, pantalla touch 7¨ a la tarjeta Jetson Nano, también conectar el módulo para conectar la red inalámbrica o también se tiene la opción de conectar el cable Ethernet para la red de Internet para visualizar la interfaz del sistema operativo instalada en lata tarjeta electrónica como se visualiza en la Fig. 85.

Fig. 85: Conexión de la interfaz gráfica y la red de internet

Elaborado por: David Chávez

3. Conectar el microcontrolador Arduino Mega 2560 a la tarjeta Jetson Nano para alimentarla. Como se observa en la Fig. 86.

Fig. 86: Conexión Arduino Mega 2560 a la tarjeta Jetson Nano

Elaborado por: David Chávez

4. Finalmente, ejecutar el sistema electrónico para que empiece a detectar el tipo de moldura, calcular la distancia de la moldura y almacenar en código QR su modelo de moldura. Como se observa en la Fig. 87, Fig. 88 y Fig. 89

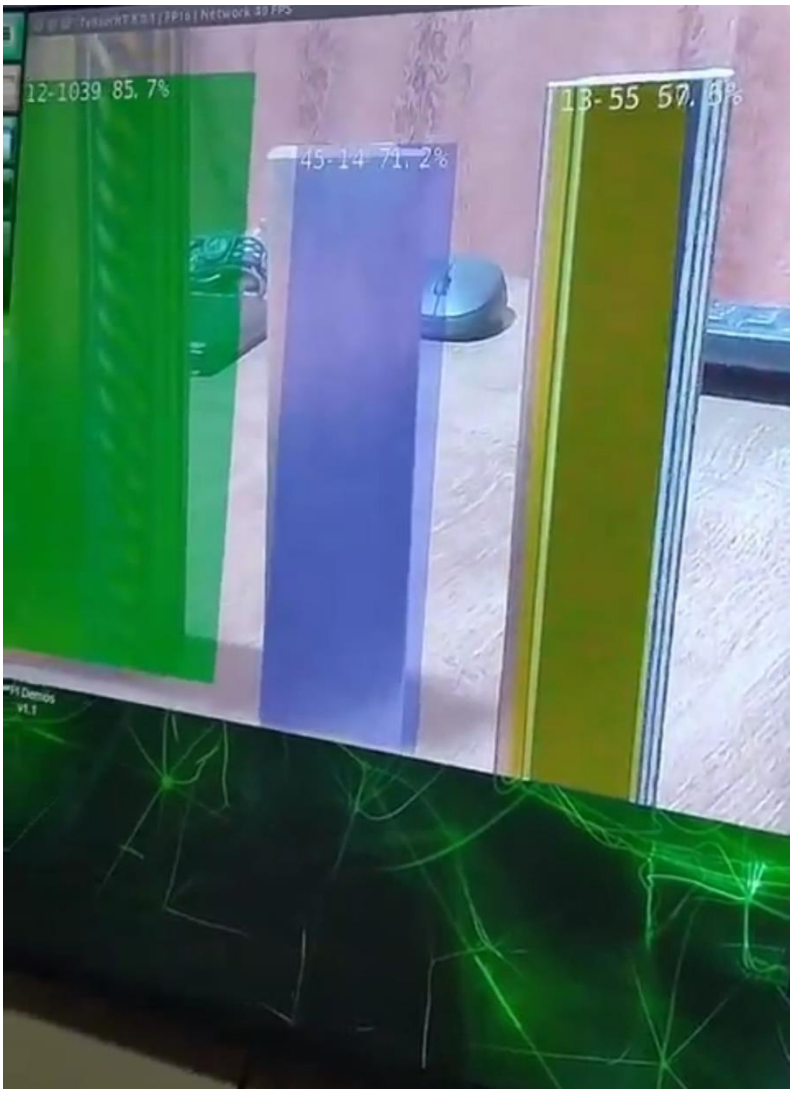

Fig. 87: Detección de los modelos de moldura

Elaborado por: David Chávez

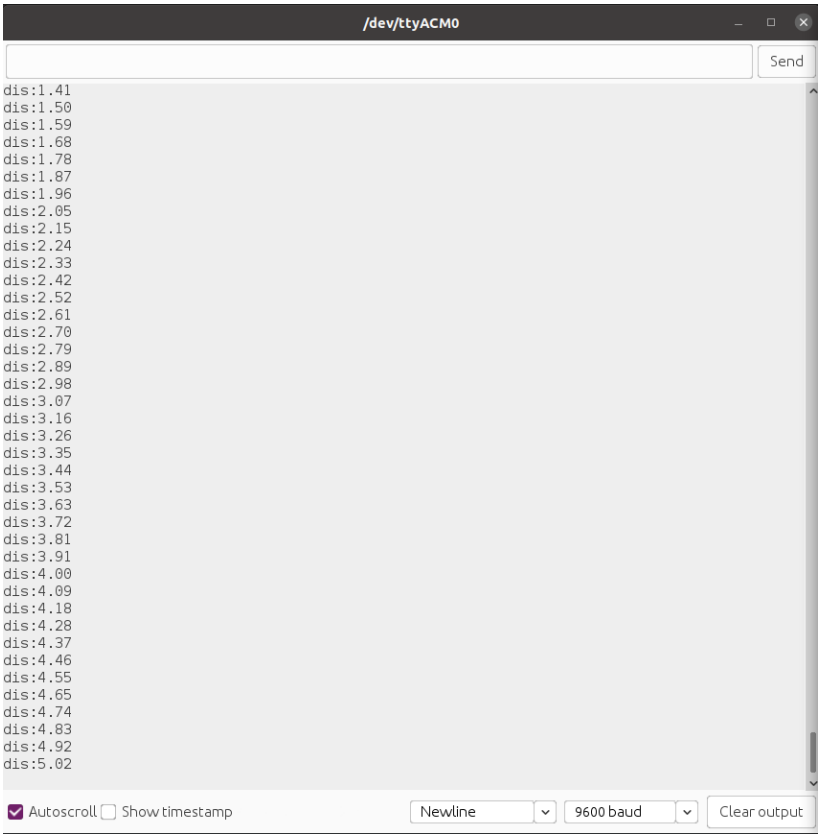

Fig. 88: Cálculo de la distancia de moldura

Elaborado por: David Chávez

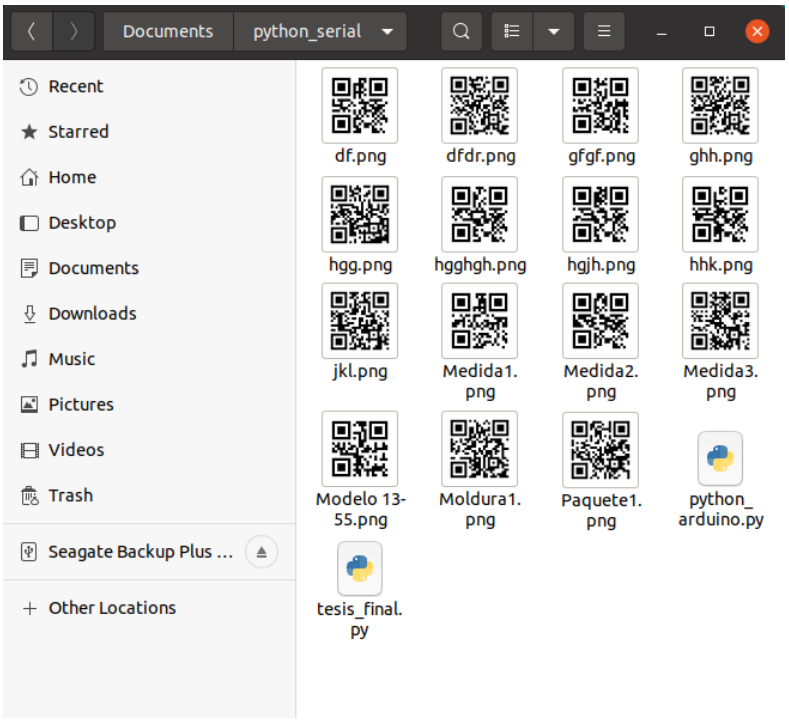

Fig. 89: Generación del código QR

Elaborado por: David Chávez