UNIVERSIDAD POLITÉCNICA SALESIANA

SEDE CUENCA

CARRERA DE INGENIERÍA DE SISTEMAS

DESARROLLO E IMPLEMENTACIÓN DE UNA PLATAFORMA DE RECONOCIMIENTO FACIAL EN TIEMPO REAL PARA LA BÚSQUEDA DE PERSONAS DESAPARECIDAS UTILIZANDO APRENDIZAJE PROFUNDO SOBRE UNIDADES DE PROCESAMIENTO GRÁFICAS

Trabajo de titulación previo a la obtención
del título de Ingeniero de Sistemas

AUTORES: ANDRÉS PATRICIO GARNICA BUENO
ANDRÉS EDUARDO USIÑA ZHINGRI

TUTOR: ING. GABRIEL ALEJANDRO LEÓN PAREDES, Ph.D.

Cuenca - Ecuador
2022

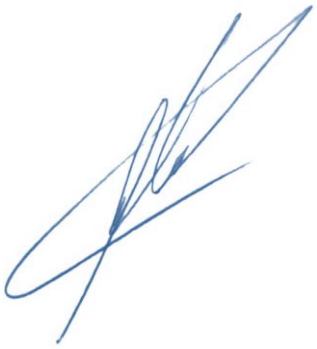
CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA DEL TRABAJO DE TITULACIÓN

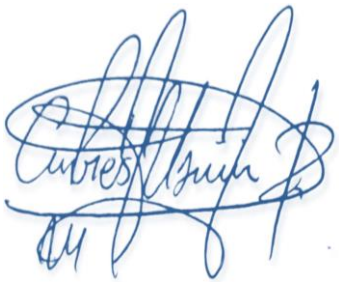
Nosotros, Andrés Patricio Garnica Bueno con documento de identificación N° 0106111180 y Andrés Eduardo Usiña Zhingri con documento de identificación N° 0106515547, manifestamos que:

Somos los autores y responsables del presente trabajo; y, autorizamos a que sin fines de lucro la Universidad Politécnica Salesiana pueda usar, difundir, reproducir o publicar de manera total o parcial el presente trabajo de titulación.

Cuenca, 24 de marzo del 2022

Atentamente,





Andrés Patricio Garnica Bueno
0106111180

Andrés Eduardo Usiña Zhingri
0106515547

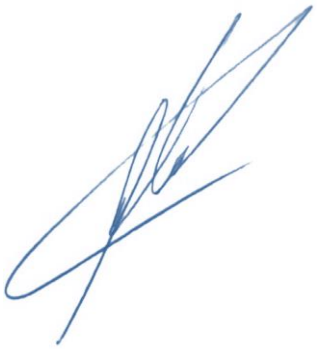
CERTIFICADO DE CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE TITULACIÓN A LA UNIVERSIDAD POLITÉCNICA SALESIANA

Nosotros, Andrés Patricio Garnica Bueno con documento de identificación N° 0106111180 y Andrés Eduardo Usiña Zhingri con documento de identificación N° 0106515547, expresamos nuestra voluntad y por medio del presente documento cedemos a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que somos autores del Proyecto Técnico: "Desarrollo e Implementación de una plataforma de reconocimiento facial en tiempo real para la búsqueda de personas desaparecidas utilizando aprendizaje profundo sobre unidades de procesamiento gráficas", el cual ha sido desarrollado para optar por el título de: Ingeniero de Sistemas, en la Universidad Politécnica Salesiana, quedado la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

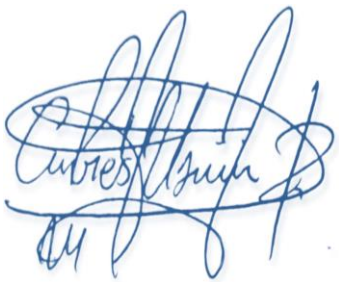
En concordancia con lo manifestado, suscribimos este documento en el momento que hacemos la entrega del trabajo final en formato digital a la biblioteca de la Universidad Politécnica Salesiana

Cuenca, 24 de marzo del 2022

Atentamente,



Andrés Patricio Garnica Bueno
0106111180



Andrés Eduardo Usiña Zhingri
0106515547

CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN

Yo, Gabriel Alejandro León Paredes con documento de identificación N° 0103652186, docente de la Universidad Politécnica Salesiana, declaro que bajo mi tutoría fue desarrollado el trabajo de titulación: DESARROLLO E IMPLEMENTACIÓN DE UNA PLATAFORMA DE RECONOCIMIENTO FACIAL EN TIEMPO REAL PARA LA BÚSQUEDA DE PERSONAS DESAPARECIDAS UTILIZANDO APRENDIZAJE PROFUNDO SOBRE UNIDADES DE PROCESAMIENTO GRÁFICAS, realizado por Andrés Patricio Garnica Bueno con documento de identificación N° 0106111180 y Andrés Eduardo Usiña Zhingri con documento de identificación N° 0106515547, obteniendo como resultado final el trabajo de titulación bajo la opción Proyecto Técnico que cumple con todos los requisitos determinados por la Universidad Politécnica Salesiana.

Cuenca, 22 de marzo del 2022

Atentamente,

Ing. Gabriel Alejandro León Paredes, Ph.D.
0103652186

AGRADECIMIENTOS

Agradezco sinceramente al personal docente y administrativo de la Carrera de Ingeniería de Sistemas por las facilidades brindadas y que con ello hicieron posible la realización de este proyecto.

De igual manera mis agradecimientos a los docentes de la Carrera de Ingeniería de Sistemas que con su conocimiento y apoyo, motivaron a desarrollarme como persona y profesional.

De manera particular, un reconocimiento al Dr. Gabriel León, Director de Tesis, quien con su conocimiento y experiencia nos dirigió durante el desarrollo de la realización de este proyecto.

Andrés Eduardo Usiña Zhingri

Agradezco primeramente a Dios por darme salud y así guiarme para poder cumplir un escalón más en la vida profesional.

Quiero agradecer a mis familiares, amigos y docentes que siempre estuvieron para inculcarnos valores, conocimientos y enseñanzas.

También agradezco al Dr. Gabriel León Paredes, por compartirnos conocimientos y darnos un apoyo incondicional en todo el desarrollo de nuestro proyecto.

Andrés Patricio Garnica Bueno

DEDICATORIA

Dedico el presente proyecto a Dios por haberme brindado la salud y la capacidad para concluir con mi objetivo.

A mis Padres quienes con su amor y su apoyo incondicional me incentivaron a nunca rendirme ante las adversidades, gracias por haberme inculcado sus virtudes y valores que hoy medefinen como persona.

A mi hermana y cuñado, Lorena y Pedro, quienes supieron aconsejarme y apoyarme en las decisiones que tomé durante el trayecto de mi vida universitaria.

A todos mis familiares y amistades que directamente o indirectamente colaboraron para que culminara con este proyecto.

Andrés Eduardo Usiña Zhingri

A mis hermanos Diego y Jonnathan "SUASANDY", que siempre han estado a mi lado apoyándome para poder culminar esta etapa de vida y así poder dar un paso más en mi formación académica. De manera especial a mis padres Patricia y Manuel, que siempre me han guiadoy aconsejado para jamás rendirme ante las adversidades que hemos tenido, sin nada de ello podría cumplir mis objetivos y mis sueños gracias. También dedicar a mis tíos, abuelos y demásfamiliares que siempre han estado apoyándome en los momentos más adversos gracias.

Andrés Patricio Garnica Bueno

RESUMEN

Ecuador ha contabilizado más de 40 casos de desapariciones que se encuentran sin resolver o llevan años de investigación a la fecha de publicación del presente documento. También este número de casos inconclusos va en aumento causando que el incremento de casos sea más difícil mitigar. Aumentar equipos de video-vigilancia o seguridad en localizaciones estratégicas puede mejorar levemente el tiempo de respuesta a casos de personas desaparecidas, pero es un método convencional que implica realizar supervisión visual constantemente. Esta investigación tiene como objetivo entrenar redes neuronales para reconocer en tiempo real el rostro de una persona especifica catalogada como persona desaparecida, usando Nvidia Digits y librerías para visión artificial para reconocer el objetivo deseado y determinar el porcentaje de precisión acertada.

Palabras Clave: Personas Desaparecidas, Nvidia Digits, Entrenamientode Redes Neuronales, Reconocimiento Facial.

ABSTRACT

Ecuador has recorded more than 40 cases of disappearances that are unsolved or have been under investigation for years as of the date of publication of this document. Also this number of unfinished cases is increasing making this increase more difficult to mitigate. Adding video-surveillance or security equipment to strategic locations may slightly improve response time to missing persons cases, but it is a conventional method that involves constant visual surveillance; This research aims to train neural networks to recognize in real time the face ofa specific person listed as a missing people, using Nvidia Digits and computer vision libraries to recognize the desired target and determine a percentage of successful accuracy.

Keywords: missing persons, Nvidia Digits, neural network training, fa-cial recognition.

Contenido

Índice de tablas

Índice de figuras

INTRODUCCIÓN

Hoy en día, existen diversos tipos de sistemas biométricos que son utilizados como medidas de seguridad, debido a que utiliza rasgos físicos únicos y distintivos de cada persona, como huella dactilar, retina o iris del ojo, etc.; con el fin de acceder o registrar un cierto tipo de información. Detrás de estos sistemas biométricos se encuentra un amplio estudio de software y hardware con múltiples funciones dirigidas a un ordenador que permite identificar a una persona. Entre estos tipos de sistemas está el reconocimiento facial ya que identifica a una persona con tan solo capturar su rostro.

En los últimos años, el reconocimiento facial ha pasado tener cierta relevancia en distintos campos con el fin de dar comodidad a usuarios en ciertas actividades requeridas por entidades o empresas. En países del primer mundo y algunos pocos de Latinoamérica usan el reconocimiento facial en aeropuertos como medida de seguridad para identificar a personas que estén siendo buscadas por la policía o la Interpol, otros utilizan el reconocimiento facial para la Educación y acceder a su información como estudiante, algunos Hospitales consideran usar el reconocimiento facial para registros e historiales médicos de sus pacientes, pero según nuestra investigación, en el Ecuador no se ha encontrado el uso de esta tecnología con el propósito de buscar personas desaparecidas. Para agravar, con la pandemia existente en el presente año, ha obligado a los países a cerrar sus fronteras y consecuencia de ello, se concentró y aumentó los casos de personas desaparecidas a comparación de las cifras que existían antes de este acontecimiento, cifras estadísticas que se mencionarán a detalle más adelante.

Por lo dicho anteriormente, la presente tesis se enfocará en el desarrollo e implementación de una plataforma de reconocimiento facial, el mismo que permitirá identificar a una persona en tiempo real que se encuentre catalogada como desaparecida acorde a los criterios establecidos por entidades competentes. Además de ello, se realizará una aplicación web para la gestión y administración de registros de datos de las personas desaparecidas, aplicación web que será optimizada y acelerada con módulos Nvidia. La finalidad de este proyecto es demostrar que su implementación contribuye y agiliza el proceso de búsquedas de personas desaparecidas a las entidades competentes.

Capítulo I

Problema

1.1. Antecedentes

Actualmente, el estado Ecuatoriano posee más de 40 casos de desapariciones de personas sin resolver o simplemente llevan años en investigación, por lo que, en la mayoría de estos casos no se ha tenido éxito para dar con el paradero de las personas [4]. En igual forma, de acuerdo a datos estadísticos proporcionados por la Dinapen, días después de que en Ecuador se declarará en estado de emergencia por COVID-19, el número de desapariciones aumentó considerablemente, tanto en casos catalogados como desapariciones voluntarias como forzadas, aunque en ambos casos su paradero es desconocido para sus allegados. [5].

Así mismo, con ayuda de medios de difusión, a la fecha actual bastantes familias solicitan que retomen pesquisas por desaparecidos, dado que estos casos desde mucho tiempo atrás fueron suspendidos sus trámites de búsquedas por diferentes acontecimientos o limitantes. Para agravar, el toque de queda impuesta por la emergencia sanitaria, paralizó la búsqueda de personas desaparecidas.

Por lo tanto, al implementar una plataforma de reconocimiento facial en tiempo real, el proceso de búsqueda de personas desaparecidas se agilizaría considerablemente, además de que familiares y allegados tendrán respuestas más rápidas al conocimiento del paradero de sus seres queridos.

1.2. Justificación

El proyecto tiene alta importancia debido a que su diseño e implementación de reconocimiento facial conjuntamente con el procesamiento de unidades gráficas, logrará reducir el tiempo de respuesta de búsqueda de una persona desaparecida, lo que permitirá a las entidades competentes agilizar su proceso de trabajo, además de que la parte fundamental de este proyecto utiliza un super-miniordenador de bajo costo, pero altamente eficiente a nivel de energía y procesamiento.

Capítulo II

Objetivos

2.0.1. General

Desarrollar una plataforma de reconocimiento facial para identificar personas desaparecidas en tiempo real basado en el entrenamiento de redes neuronales profundas sobre unidades de procesamiento gráficas

2.0.2. Específicos

- Realizar un análisis de las técnicas de visión por computadora utilizadas para la detección de rostros en tiempo real.
- Evaluar y Seleccionar los mejores modelos del estado del arte para detección de rostros de personas desaparecidas.
- Entrenar y validar modelos para la detección de rostros de personas desaparecidas mediante redes neuronales profundas de aprendizaje.
- Desarrollar una aplicación para la gestión y administración del sistema de personas desaparecidas.
- Realizar y Desplegar modelos entrenados en equipos de hardware de cómputo de alto rendimiento usando cámaras de alta visión en tiempo real.
- Realizar pruebas de funcionamiento y rendimiento de la plataforma de reconocimiento facial.

Capítulo III

FUNDAMENTOS TEÓRICOS

3.1. Inteligencia Artificial

3.1.1. Definición

La inteligencia artificial (IA) en la actualidad es popular gracias a los grandes volúmenes de datos, o algoritmos que se manejan a nivel mundial, así mismo, en sus inicios esta se centró solo a la solución de problemas y métodos simbólicos, a causa de esto en los años 60 el departamento de Defensa de los Estados Unidos se mostró interesado en explorar esta nueva tecnología porlo que empezó a entrenar computadoras para que imitaran el razonamiento humano.

De manera que la inteligencia artificial es la combinación de algoritmos entrenados con el propósito de crear máquinas que presenten las mismas capacidades o destrezas que el ser humano, como por ejemplo computadoras que juegan ajedrez o automóviles de conducción autónoma.

3.1.2. Tipos

Según las investigaciones propuestas por los autores Martínez, Dalgo y Herrera [6], detallan que existen cuatro tipos de Inteligencia Artificial(IA):

- **Máquinas reactivas:** Son sistemas de cómo piensan los humanos, es decir son máquinas diseñadas para automatizar actividades en la toma de decisiones, por lo que este tipo de IA no almacena ni memoriza solo analiza la información en un tiempo determinado para la resolución de problemas.
- **Máquinas con memoria limitada**: En relación a este tipo de IA son sistemas que actúan como humanos, es decir, son computadoras que efectúan las tareas de forma similar a como lo hacen las personas. Como es el caso de los robots.

- **Máquinas con teoría de la mente:** Acerca de este tipo de IA son sistemas que piensan racionalmente, en otras palabras, estos intentan emular el pensamiento lógico racional de los humanos, es decir, con esto se investiga cómo lograr que las máquinas puedan sentir, razonar y actuar en consecuencia.
- **Maquinas con autoconciencia:** Son sistemas que actúan racionalmente, es decir, son máquinas capaces de imitar de manera racional el comportamiento humano, como percibir,razonar y actuar.

3.1.3. Aplicaciones

Muchas de las herramientas de software o aplicaciones que se utiliza día a día ya sea por cuestiones laborales o de entretenimiento utilizan inteligencia artificial. En los siguientes párrafos se detallan de manera concisa sus usos más destacados:

- **Machine Learning:** El Aprendizaje Automático consiste en una disciplina de las ciencias informáticas, relacionada con el desarrollo de la Inteligencia Artificial, y que sirve, como ya se ha dicho, para crear sistemas que pueden aprender por sí solos. Es una tecnología que permite hacer automáticas una serie de operaciones con el fin de reducir la necesidad de que intervengan los seres humanos. Esto puede suponer una gran ventaja a la hora de controlar una ingente cantidad de información de un modo mucho más efectivo [7].
- **Deep Learning:** El Deep Learning o aprendizaje profundo se define como un algoritmo automático estructurado o jerárquico que emula el aprendizaje humano con el fin de obtener ciertos conocimientos [8]. Una cualidad de ésta es que se caracteriza por no requerir reglas previamente programadas, es decir, el propio sistema tiene la capacidad de «aprender» de manera autónoma para ejecutar una tarea por medio de una fase previa deentrenamiento. Otra característica que la destaca es que está compuesto por redes neuronales artificiales entrelazadas utilizadas para procesar la información; que se emplea principalmente para la automatización de análisis predictivos .
- **Inteligencia Artificial en otras áreas:** Dado que la tecnología está en un avance constante; el usuario utiliza con más frecuencia aplicaciones con IA sin darse cuenta, aplicacionesque van beneficiando a varias áreas, algunos de estos ejemplares son:
    - Finanzas
    - Salud
    - Seguridad
    - Educación
    - Comercial
    - Agrícolas
    - Climáticas
    - Logísticas y Transporte

3.2. Visión Por Computadora

3.2.1. Definición

La visión por computadora es un conjunto de tecnologías que consiste en la extracción automatizada de la información de imágenes a través de computadoras, robots, drones, etc. Además, éstas a menudo se emplean para realizar operaciones como reconocimiento de objetos. En resumen, esta tecnología se encarga en proporcionar a los dispositivos antes mencionados a reaccionar dependiendo de la información que obtiene.

Con respecto a la visión por computadora, esta es similar a la Inteligencia Artificial ya que contiene una mezcla de programación modelada y matemática puesto que se convierte en un campo llamativo y atractivo para los investigadores, debido que en la actualidad este trata en imitar a la visión humana, pero a su vez se enfoca más para resolver un problema específico.

3.2.2. Aplicaciones

La visión artificial es una tecnología que ha estado teniendo bastante aceptación en diferentes campos por el ahorro de tiempo que ésta ofrece, mejorando calidad y costes según su aplicación. A continuación, se detalla algunas de las aplicaciones que tiene la visión artificial:

- **Reconocimiento óptico de caracteres (OCR):** Consiste en la identificación automáticamente a partir de una imagen de símbolos o caracteres que pertenecen a un determinado alfabeto, para luego almacenarlos en forma de datos.
- **Inspección robotizada:** Trata de la inspección rápida de las piezas para garantizar la calidad de los componentes de fabricación utilizando una visión estéreo con iluminación especializada.
- **Venta al por menor:** Encontramos los lectores de barras para reconocer los precios de los productos en la línea de cajas en los supermercados.
- **Construcción de modelos 3D**: La construcción automatizada de modelos 3D a partir de fotografías.
- **Imágenes médicas:** Se usa en la toma de radiografías y para detectar tumores malignos y anomalías en las mismas.
- **Seguridad automotriz:** Ayudando a detectar obstáculos mediante un sistema de conducción asistida utilizando diferentes cámaras.
- **Captura de movimiento:** Utilizando marcadores retro-reflexivos vistos desde múltiples cámaras u otras técnicas para la captura de movimientos de los actores para utilizar en animación por computadora.
- **Vigilancia:** Monitoreo de intrusos, análisis del tráfico vial, y monitoreo de piscinas para víctimas de ahogamiento.

- **Reconocimiento de huellas dactilares y biometría:** Para la identificación automática de accesos y también utilizada para aplicaciones forenses.
- **Detección de rostros:** Utilizado para mejorar el foco de las cámaras y para hacer una búsqueda más relevante de personas en imágenes.

3.3. Reconocimiento Facial

3.3.1. Definición

El reconocimiento Facial es una de los diversos tipos de tecnología que utilizan los sistemas biométricos para identificar o verificar la identidad de una persona en función de sus características faciales (ojos, nariz, pómulos, frente, barbilla, entre otros). En el área técnica, detrás del proceso de reconocimiento facial contribuye más estudios minuciosos ideales para el correcto funcionamiento como reconocimiento de patrones y relieves, óptica, visión artificial e incluso conceptos esenciales de la estadística. [9]

3.3.2. Métodos de Detección Facial

El proceso para la detección de rostros es bastante complejo y exigente, ya que sus resultados pueden variar por diversos factores sin importar el método que se le haya implementado. Según los autores Espinoza y Jorquera, los métodos de detección facial se pueden basar acorde a los siguientes criterios [10]:

- **Basados en conocimiento:** Codifican el conocimiento humano mediante distancias y posiciones entre las características humanas (ojos, nariz, labios).
- **Basados en características invariantes:** son aquellas que no se modifican a eventuales cambio de luz, pose o ubicación de la cámara, tales como la ceja, nariz, textura de la piel y línea de pelo. Funciona detectando uno de estos componentes, construyendo un modelo estadístico y con los resultados, verificar la existencia de un rostro.
- **Basados en moldes (patrones):** Es la relación entre una imagen de entrada y un patrón o molde previamente definido, cuyo objetivo es capturar características del rostro.
- **Basados en apariencia:** Utilizan modelos obtenidos mediante entrenamiento de imágenes, tomando la imagen como un vector de características, es decir, es visto como una variable aleatoria. A diferencia de los métodos basados en moldes, donde el patrón es de- finido por un "experto", los patrones en este modelo son determinados por el aprendizajeobtenido en el entrenamiento de imágenes.

3.3.3. Tipos de Reconocimiento Facial

El reconocimiento facial puede operar de diferentes formas según la necesidad de la empresa que adquiere este tipo de tecnología, pero en su mayoría simplemente compara una imagen del rostro desconocido con todas las imágenes de rostros almacenadas en una base de datos, donde

su sistema confirmará su identidad. Según el tipo de equipamiento que utilice el reconocimiento facial se le puede clasificar de dos maneras:

3.3.3.1. Reconocimiento facial 2D

El funcionamiento de reconocimiento facial en 2D radica en tomar puntos de referencia del rostro, como la nariz, ojos, boca, rasgos faciales más fáciles de identificar a una cierta distancia, ya que a partir de ellos realiza mediciones del ancho y forma de dichos rasgos que posteriormente son transformadas en codificación numérica a través de softwares para encontrar coincidencias. [11]. Sin embargo, si la cámara es de gama baja no logrará distinguir si lo que está capturando es una imagen de rostro o una fotografía con un rostro, debido a que se ve afectada también por la cantidad de iluminación, variaciones de ángulos con respecto a la cámara; siendo una desventaja a la seguridad por ser considerado ineficiente.

3.3.3.2. Reconocimiento facial 3D

Con la implementación de cámaras de gama alta, sus complementos llegan a incluir tecnología infrarroja combinada con la tecnología 3D, mejorando la eficiencia del reconocimiento facial. Algunas cámaras de este tipo detectan profundidad de los ojos, distancia entre boca y nariz por lo cual el uso de accesorios que oculten rasgos fisiológicos o fotografías alteradas noserán un problema, mejorando la identificación de la persona de manera segura y rápida. [12].

3.3.4. Aplicaciones

Actualmente, el reconocimiento facial es una tecnología que todavía está en desarrollo, es decir, que todavía no existe una versión 100 % estable, ni siquiera en países de primer mundo, pero eso no ha sido impedimento para encontrarles múltiples usos que, de cierta forma, mejoran la calidad de vida como sociedad en distintas empresas o entidades como por ejemplo Institutos Educativos, Aeropuertos, Hospitales, Empresas que requieran que solo usuarios autorizados tengan acceso a espacios determinados, entre otros. Por otra parte, el tema de reconocimiento facial llega a ser bastante debatible ya que presenta tanto ventajas y desventajas que son bastante cuestionados. A continuación, se detallan algunas ellas:

Ventajas	Desventajas
Protección de dispositivos y objetos personales. Agilidad en trámites aeroportuarios Seguridad en trámites bancarios Alta precisión en detección	Privacidad de Datos cada vez disminuye. Requiere Cámaras de gama alta Almacenamiento de datos requiere espacio

Tabla 3.1: Ventajas y Desventajas del Reconocimiento Facial

3.4. Unidad de Procesamiento Gráfico (GPU)

3.4.1. Definición

Actualmente, una gran cantidad de sistemas actuales de computación posee como parte fundamental la GPU, un componente de hardware considerado como un coprocesador que permiterealizar operaciones complejas o procesamiento de gráficas de un sistema de cómputo, con el fin de aligerar la carga de trabajo del CPU, especialmente en aplicaciones como videojuegoso aplicaciones 3D Interactivas. Esto es posible gracias a que estos componentes "disponen de un número alto de núcleos, mejor denominados como *cores* con un alto ancho de banda con memoria, permitiendo aumentar la capacidad de procesamiento de las CPU". [13]

3.4.2. Computación de Propósito General en GPU (GPGPU)

En un inicio, las GPU procesaban sus gráficas a través de funciones fijas, pero con el avance del tiempo, los investigadores en esta área descubrieron que la GPU podría aplicarse en operaciones de cálculo de punto flotante en las que demandaba un aumento de velocidad en ejecución, optimizando diversas aplicaciones utilizadas en la comunidad científica, dando origen a un nuevo concepto llamado GPGPU o GPU de Propósito General.

Corporaciones como Nvidia y AMD, han orientado el uso de la GPU a nuevos tipos de aplicaciones, dando soluciones a distintos problemas de procesamiento como paralelismo a nivel de datos, aprovechando al máximo la potencia de hardware de la GPU y resolver operaciones complejas por medio de programación de alto nivel. [14]

En otras palabras, el término GPGPU consiste en la ejecución de procesos que típicamente eran pensadas para la CPU, pero aprovechando el máximo potencial de la GPU, cuyos procesos eran tareas altamente paralelizables. Para la facilitar estas tareas, existen dos marcos populares: CUDA de NVidia y una implementación de código abierto OpenCL, permitiendo así el acceso a la tarjeta gráfica. Por consiguiente, cualquier usuario puede escribir su propio código de programa utilizando la GPU para cálculos de propósito general y no solamente a limitarse con el procesamiento gráfico. Desde otro punto de vista, la GPU realiza operaciones relacionadas solo a gráficos mientras GPGPU hace referencia a gráficos más su propia unidad de programación o cualquier tarea que pueda ser paralelizada.[13]

3.5. Nvidia CUDA

La evolución constante de la computación permite dividir tareas grandes en cosas más pequeñas para poder realizarlas todas estas al mismo tiempo. Actividades como los juegos en línea, renderización de videos, o incluso softwares para programación móvil requerían una gran cantidad de recursos en cuanto a capacidad de procesamiento, pese a que algunos ordenadores tenían los mejores y últimos procesadores en el mercado, seguía dificultando la eficiencia en sus

actividades. Es aquí donde la Corporación Nvidia, propone la iniciativa de dividir y repartir la carga de trabajo de tareas entre el procesador y las GPU a través de una arquitectura de cálculo en paralelo a la que denominaron CUDA.

3.5.1. Definición

CUDA (Arquitectura Unificada de Dispositivos de Cómputo) es tecnología moderna de computación paralela y modelo de programación que permite utilizar la GPU de la tarjeta gráfica en tareas que antes estaban siendo procesadas por la CPU. Es decir, que con la ayuda de CUDA, la GPU es otro procesador que se dedica de manera exclusiva al procesamiento de gráficos para aligerar y balancear la carga de trabajo del procesador central, como es en el caso delos videojuegos y aplicaciones 3D, esta es la razón por la cual existen grandes cantidades de núcleos CUDA en las GPU. [15]

3.5.2. Arquitectura y Funcionamiento

La Arquitectura de una GPU suele basarse en el modelo circulante, un tipo de arquitectura que facilita el procesamiento en paralelo y la segmentación para realizar sus respectivas tareas. Por otra parte, la Corporación Nvidia ha decidido optar por otro tipo de arquitectura al momento de diseñar sus propias GPUs, que es la arquitectura Pascal; conformada por millones de transistores e incluidos con más de 2500 núcleos que, combinadas con herramientas desarrolladas porla Corporación Nvidia mejoran el procesamiento y consumo de energía.

Hecha la observación anterior, la arquitectura Nvidia CUDA está basada en tarjetas gráficas de Nvidia que tiene como objetivo a realizar cálculos en paralelo; pero su fortaleza radica en sus GPUs que puede tener miles de núcleos, a los que se denomina *CUDA Cores o núcleos CUDA*. Cada núcleo CUDA es un procesador paralelo bastante similar a un procesador de ordenador de dos o cuatro núcleos, con la diferencia de que ejecutan varias tareas que a su vez pueden subdividirse para obtener resultados independientes y al final volver a unirlos y presentarlo como un solo resultado, permitiendo que el número de núcleos esté relacionado directamente a lavelocidad y potencia de la GPU.

Para entender de mejor manera la arquitectura Nvidia CUDA, en la figura 2.1, el autor Solé [1] explica a detalle el funcionamiento de los *CUDA Cores*. En la imagen se muestra 8 unidades de ejecución conocidas como *Streaming Multiprocessors (SM)*, que se encuentra interconectadas entre sí por una zona de memoria en común. En la figura 2.2, cada SM está compuesto por varios núcleos de cómputo llamados núcleos CUDA, que son responsables de la ejecuciónde instrucciones, en este caso existe 32 núcleos por cada SM, totalizando en 256 núcleos de procesamiento.

Figura 3.1: Detalles de la arquitectura Nvidia GF100 [1]

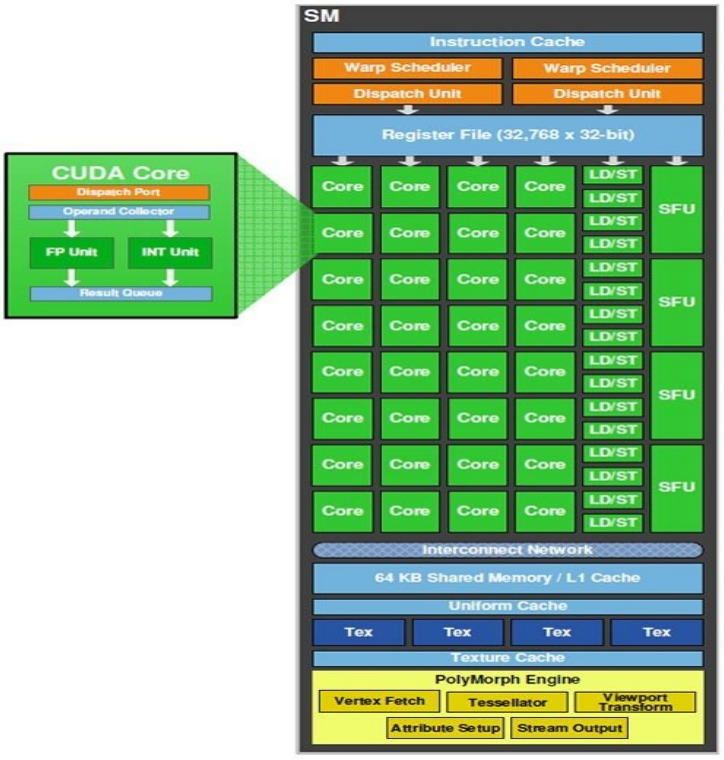


Figura 3.2: Streaming Multiprocessors (SM) [1]

Este diseño puede ser considerado como un tipo de arquitectura a nivel de hardware, permitiendo programar los núcleos de la GPU usando lenguaje de alto nivel como bien puede ser Python o C para Nvidia CUDA. Ahora un programador puede escribir un programa secuencialen la que internamente se le conoce como Kernel, y puede ser desde una función sencilla has- ta un programa completo. Una vez ejecutado el Kernel se lo procesa en paralelo dentro de la GPU en un conjunto de hilos o *threads*, en la que el programador puede previamente organizardentro de una jerarquía agrupándolas en bloques o *blocks* y a su vez son distribuidas en gruposformando una malla o *grid*.

Resumiendo lo planteado, en la figura 2.3 se aprecia como trabaja la Arquitectura CUDA; un bloque es un conjunto de hilos simultáneos que trabajan entre sí de manera sincronizada y comparten espacio de memoria exclusivo de cada bloque, mientras que la malla es un conjunto de bloques que son ejecutados de manera independiente y lanzados en paralelo en los Streaming Multiprocessors (SM). Al momento de invocar un kernel, el programador especifica el número de hilos por bloque y número de bloque que conforma la malla para que sean procesados dentro de la GPU, cada hilo es asignado a un ID único dentro de su bloque, y cada bloque recibe un ID único dentro de la malla, al hacer esto simplifica el direccionamiento de memoria al trabajar con datos multidimensionales, dando un tiempo de respuesta rápido al kernel, un claro ejemplo de ésta es el procesamiento de imágenes.

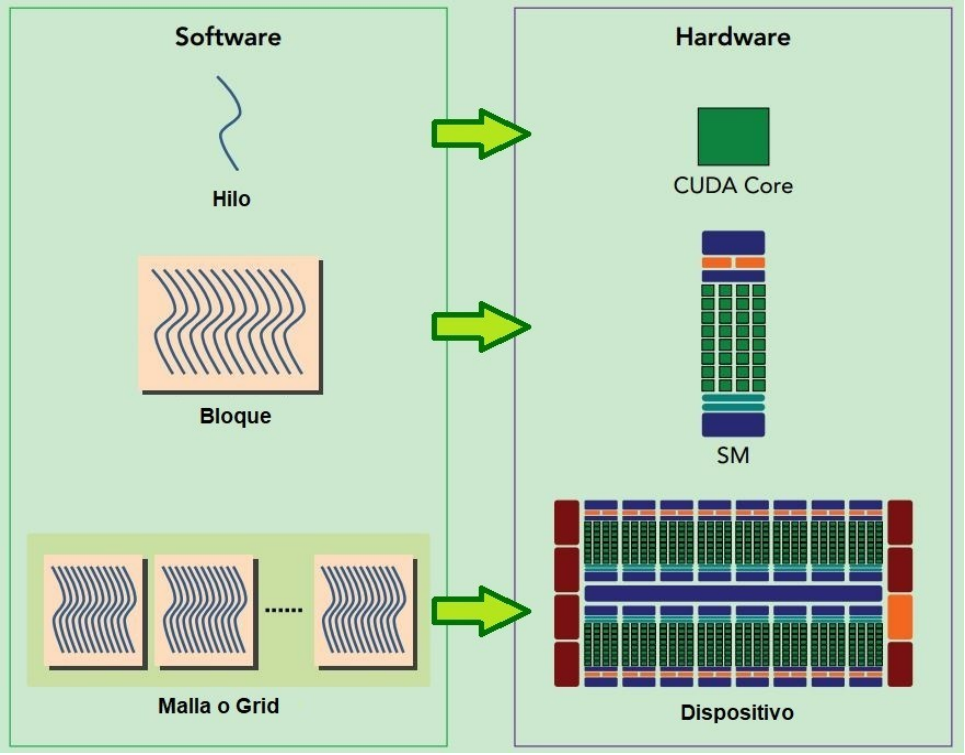


Figura 3.3: Funcionamiento de Arquitectura CUDA

3.5.3. Beneficios

Para usuarios que diseñan o trabajan en edición de video, CUDA acelera la velocidad para procesamiento gráfico, dando una alta calidad de definición a la renderización de videos. Cuando la GPU realiza operaciones gráficas se le denomina primitivas, que posteriormente son optimizadas para mejorar la calidad de la imagen. Entre las primitivas más conocidas que ofrece CUDA en el procesamiento de gráficas en 3D, es el *Anti-aliasing*, una función que suaviza los bordes de las figuras para dar un mayor rendimiento y dar efectos más realistas. Nvidia CUDA. puede acelerar el procesamiento hasta 10 veces más si se trabaja con softwares que soporten la tecnología CUDA, un claro ejemplar es XiliSoft.

Aunque CUDA ha dado soluciones gráficas en el campo de videojuegos, sus aplicaciones van más allá. CUDA es muy valorada por desarrolladores y científicos, pues permite trabajar con deep learning, Ingeniería y Análisis de datos usadas por Universidades, entidades Gubernamentales, Empresas PYMES y grandes a nivel global. La página oficial de NVidia. muestrauna amplia lista de campos que utilizan CUDA, La lista completa se encuentra en la sección ***Aplicaciones aceleradas por GPU***, [16], pero a nuestro criterio, entre las más destacadas están:

- **Ingeniería Asistida por Computadora:** Abaqus/Standard, Nastran, ANSYS Mechanica
- **Defensa e Inteligencia:** DigitalGlobe Advanced Ortho Series, Incogna GIS, MotionDSP Ikena ISR.
- **Medios y Entretenimiento:** Adobe CS6, Autodesk 3ds, Telestream Vantage.
- **Investigación Petrolera:** Acceleware AxRTM, ffA SVI Pro, Headwave Suite, Western-Geco Omega2 RTM
- **Cómputo Científico:** MATLAB, Chroma, AMBER, CHARMM, GAMESS, GROMACS, QUDA.
- **Pronóstico de Tiempo:** COSMO, GEOS-5, HOMME, HYCOM, WRF, NEMO, NIM

3.6. Nvidia DIGITS

3.6.1. Definición

Nvidia DIGITS, de sus siglas en Inglés: Nvidia **D**eep Learning **G**PU **T**raining **S**ystem; es una aplicación web que permite el entrenamiento de modelos de aprendizaje profundo de forma rápida y en alta precisión (*Deep Learning*), diseñada especialmente para tareas de clasificación de imágenes, segmentación y detección de objetos. [17]

3.6.2. Características

- Permite diseñar, entrenar y visualizar redes neuronales profundas.

- Posee algunos modelos pre-entrenados como AlexNet, GoogLeNet, LeNet y UNET de DIGITS Model Store.
- Permite la programación, monitoreo y administración de trabajos con Deep learning en tiempo real.
- Posee alta variedad de formatos y fuentes de imágenes con el complemento de DIGITS
- Escala los trabajos de capacitación en múltiples GPU automáticamente

3.6.3. Requerimientos para la plataforma Nvidia DIGITS

GPU de Entrenamiento	Despliegue
Instancia de TITAN, Quadro, Tesla o NGC basada en Maxwell, Pascal o Volta Ubuntu version 14.04 o 16.04 para 64 bits	Kit de desarrollador Jetson TX1 con JetPack 2.3 o posterior (Ubuntu 16.04). Kit de desarrollador Jetson TX2 con JetPack 3.0 o posterior (Ubuntu 16.04). Jetson AGX Xavier Developer Kit con Jet-Pack 4.1 Developer Preview Acceso anticipado (Ubuntu 18.04).

Tabla 3.2: Requerimientos del Sistema para DIGITS

3.7. Jetson TX2

En realidad, se trata de un módulo de Nvidia que optimiza y agiliza la carga de trabajo durante el procesamiento de gráficas. Jetson TX2 es un mini-superordenador de inteligencia artificial (IA) basada en la arquitectura Nvidia Pascal. Su función es ofrecer el máximo rendimiento posible y eficiencia energética para la carga de trabajo con cálculos altamente exigentes.Esto permite al módulo concentrar una enorme cantidad de potencia en un formato pequeño debajo consumo; es similar al tamaño de un Raspberry Pi pero con la diferencia de que su rendimiento es superior en cuanto a características técnicas como procesador, GPU, almacenamiento,interfaces, etc. A continuación, se detalla una tabla comparativa entre las dos tecnologías. [18]

	Raspberry Pi 3 Model B+ Fundación Raspberry Pi	Jetson TX2 NVIDIA
		
SoC	Broadcom BCM2837B0	NVIDIA Tegra X2 "Parker"
Especificaciones del procesador primario	ARM Cortex-A53 (64-bit) 1.4GHz quad core	4x ARM Cortex-A57+ NVIDIA Denver2 (dual core) (64-bit) 2GHz Hexa-core
GPU	Video Core IV	Nvidia Pascal
Tamaño de RAM	1 GB	8 GB
RAM incorporada	✓	✓
Almacenamiento interno	X	32 GB
Almacenamiento extraíble	Micro SD	Otro Formato
Puerto USB	4	2
USB OTG	✓	X
Versión de USB	2.0	3.0 + 2.0
Ethernet	1 puerto Tipo: Gigabit	1 puerto Tipo: Gigabit
HDMI	✓	✓
VGA	X	X
Video compuesto (CVBS)	✓	X
Interfaz de pantalla	DSI, combo 3.5mm A/V Jack	2x MIPI DSI
Interfaz de cámara	1	1
Salida de audio	✓	X
Conector de audio	3.5mm jack	X
HDMI audio	✓	✓
SATA	X	X
Sensor IR	X	X
WiFi	802.11 b/g/n/ac	802.11 a/b/g/n/ac 2x2 867Mbps
Bluetooth	Bluetooth BLE 4.2	Bluetooth 4.1
Potencia de entrada , voltaje	4.8 V - 5.2V	5.5V - 19.6V
Potencia de entrada, corriente	600mA - 2.4A	810mA -2.5A
Soporte de Linux	Raspbian, etc.	Si

Figura 3.4: Jetson TX2 vs RasberryPi

A nivel comercial, Nvidia Jetson TX2 se ha propuesto como solución a problemas de rendimiento en el campo de la inteligencia artificial como sustituto a cerebros de robots, drones, dispositivos médicos y en cámaras con IA para ciertos sectores de las ciudades. Un claro ejemplo de este último se aprecia en la figura 2.5 y figura 2.6, proyectos simples desarrollados por usuarios aplicando reconocimiento facial y reconocimiento de objetos con una Jetson Nano, cuyas características técnicas son bastantes similares a la Jetson TX2 pero de menor capacidad. Ambos módulos logran ejecutar redes neuronales más grandes y profundas perfeccionando dispositivos más inteligentes como son el caso de las cámaras, mejorando el tiempo de respuesta y precisión a la clasificación de imágenes o reconocimiento de objetos que estas captan.

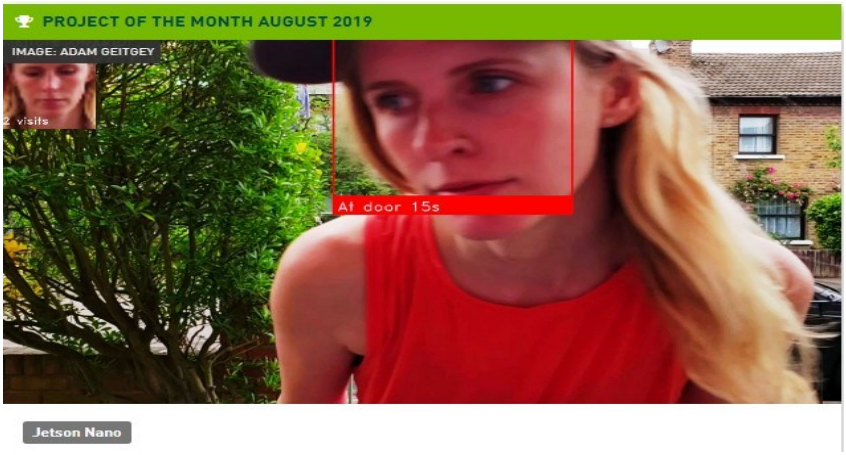


Figura 3.5: Reconocimiento Facial con Jetson Nano. [2]

Figura 3.6: Reconocimiento de Objetos con Jetson Nano. [3]

3.8. Trabajos Relacionados

Autores de un artículo publicado en la universidad de la República, en Uruguay, presentaron una solución a uno de los problemas que tiene el reconocimiento facial al momento de su funcionamiento, que hace referencia hacia donde apunta la cara de una persona con respeto a una determinada imagen, en la que sus autores aplican el procesamiento en paralelo de la GPU con ayuda de las herramientas que ofrece Nvidia CUDA, para el entrenamiento y evaluación de redes neuronales y así resolver al problema planteado anteriormente. Sus resultados fueron bastante satisfactorios, obtuvieron una reducción significativa de tiempos de cómputo en el procesamiento de imágenes. "*Speedup* mayores a 8 se obtuvieron al contrastar una implementaciónen paralelo con una secuencial y tasas de clasificación mayores a 85 % son obtenidas". A travésde análisis experimental con un algoritmo de retro propagación obtuvieron mejoras en tiemposde ejecución. Demostrando que la potencia de cálculo de la GPU puede utilizarse en procesamiento de imágenes de tamaños considerables con una resolución de hasta 240 pixeles. [19]

En la universidad autónoma de occidente de Cali-Colombia, expuso un artículo acerca de la clasificación de imágenes usando el *Deep Learning* con ayuda de frameworks y librerías más conocidas y utilizadas en los últimos tiempos: TensorFlow, Pytorch y Caffe, usando a su vez una CNN de arquitecturas AlexNet y ResNet para el reconocimiento de imagénes cuya clasificación se ejecuta en un sistema embebido poco tradicional otorgado por la corporación Nvidia,una Jetson Tx2. Al final de su proyecto ofrecen diversas comparativas de los resultados obtenidos, demostrando que Caffe es más rápido al acceder a datos que TF y Pytorch, por usar unabase de datos tipo llave valor; la velocidad de entrenamiento con AlexNet con el uso de la GPUes 22x veces más rápido a diferencia del mismo entrenamiento de red con el uso de CPUs. La velocidad de entrenamiento con redes AlexNet y ResNet usando CPUs Core i5 en portátiles y PCs de Escritorio, es relativamente igual al usar una Jetson TX2, con la diferencia que con estaúltima da una ventaja en consumo de energía, ya que el procesador de la Jetson TX2 consume 7.5 veces menos energía con respecto a la CPU de PCs de escritorio y 1.5 veces inferior en relación a las CPUs de portátiles, concluyendo que la Jetson TX2 es una mejor opción para serutilizada en plataformas portátiles. [20]

De igual manera, un proyecto similar en la universidad autónoma metropolitana, sus autores proponen la clasificación de imágenes utilizando redes neuronales convolucionales utilizando herramientas de Nvida CUDA con las librerías de TensorFlow, cuya finalidad era proponer una arquitectura eficiente para la clasificación de imágenes debido a que hoy en día no existe un método 100 % fiable para cumplir dicho propósito; además de mostrar las ventajas y desventajas de su arquitectura propuesta a comparación de otras redes mundialmente conocidas como AlexNet, GoogleNet y ResNet. Sus Resultados demostraron que la arquitectura propuesta por los autores eran eficientes en un 90 % al utilizar menos capas para extracción de características y su tiempo de entrenamiento era menor con respecto a ResNet, pero de mayor tiempo de entrenamiento que AlexNet y GoogleNet. [21]

Evidentemente, en la mayoría de trabajos relacionados al procesamiento de imágenes para su reconocimiento, utilizan las herramientas desarrolladas por la Corporación Nvidia, CUDA, ya que proporciona un kit de herramientas llamado CUDA toolkit, ideal para desarrollar, optimizar e implementar aplicaciones en sistemas integrados acelerados por GPU. [22]. Una ventajade CUDA toolkit es que dispone de librerías para acelerar aplicaciones personalizadas que hayan sido programadas en lenguajes como C, C++, Fortran, Python, MatLab; siendo de gran utilidad para desarrolladores que tengan estos lenguajes en preferencia.

Con base a los trabajos relaciones analizados en los párrafos anteriores, es posible identificar a una persona con mayor precisión utilizando módulos de Nvidia, las cámaras Jetson TX2 de Nvidia y el resto de sus componentes en total poseen el tamaño de una tarjeta de crédito, por ende, no requiere de bastante espacio físico, además de que consumen menos energía que su antecesor. Sus interfaces para dispositivos periféricos son útiles y compatibles con USB y Ethernet, lo que le hace ideal para conectar cualquier dispositivo inteligente.

En definitiva, luego de realizar toda esta investigación a profundidad, se ha propuesto que nuestro proyecto trabajará con herramientas que permitan trabajar el procesamiento de imágenes conjuntamente con la programación, que en este caso será Nvidia DIGITS. Esta aplicación web permite entrenar múltiples modelos de Deep learning, lo que facilita las

multitareas conel hardware. Además de ser compatible con algunos lenguajes de programación, entre ellos C++ o Python, lenguajes utilizados en algoritmos para reconocimiento facial que llevará nuestro proyecto. Por último, el establecer y proporcionar patrones previamente a Nvidia DIGITS, al igual que la selección de los mejores modelos de Deep Learning, contribuyen de gran manera al reconocimiento facial, cumpliendo los primeros objetivos propuestos a nuestro proyecto.

Capítulo IV

MARCO METODOLÓGICO

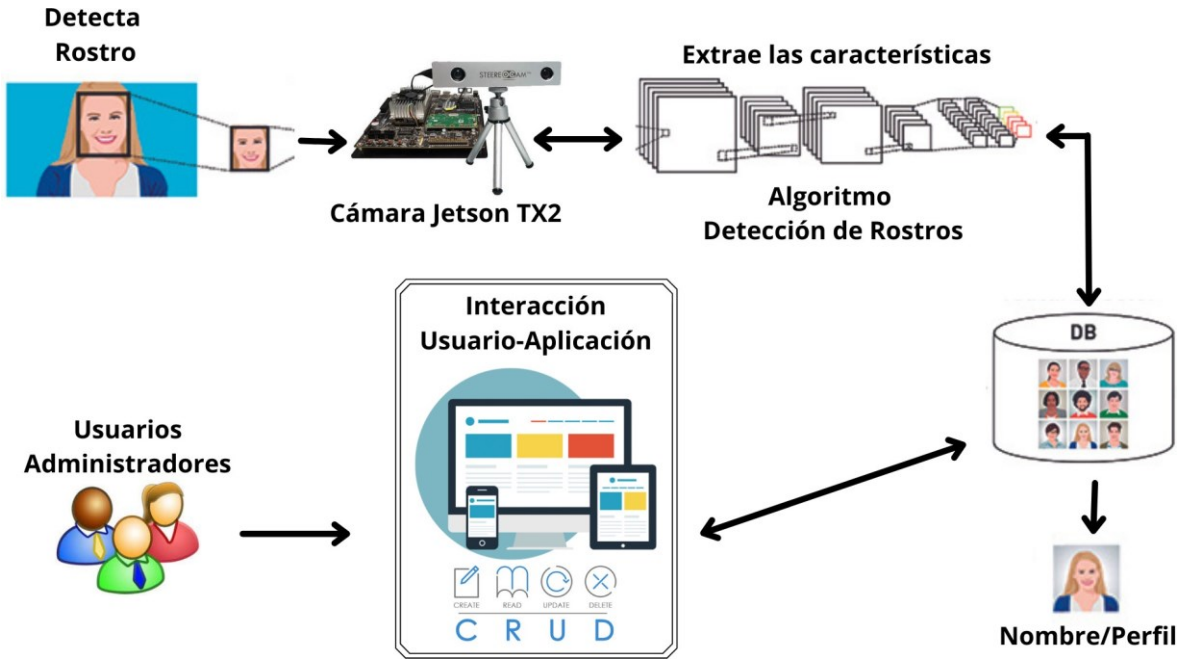


Figura 4.1: Arquitectura de la Aplicación Web

Para este proyecto se creó una Red Neuronal Profunda (Deep Neural Netowrk o DNN por sus siglas en Inglés), la cual se personalizó para que este enfocado a la búsqueda de un rostro catalogado en este caso como Persona Desaparecida. La DNN se crea bajo un contenedor que ofrece la plataforma de Nvidia, una herramienta que permite la creación y entrenamiento de archivos de modelos. El archivo generado por esta herramienta contiene todos los datos obtenidos de las imágenes previamente seleccionadas para posteriormente cargar este archivo a la Jetson TX2 e iniciar con el proceso de clasificación de imágenes considerado el tipo de objeto en la imagen, capas de colores RGB y profundidad; datos que serán interpretados y procesados por la misma Jetson TX2, que gracias a sus múltiples núcleos de GPU que posee, puede procesar cálculos complejos para clasificar las imágenes y mostrar los resultados con un alto porcentaje

acertado. Además, para una mejor interacción con el usuario se despliega una aplicación web en un servidor que permite realizar todas las actividades anteriores descritas al igual que recibir dichos resultados de manera intuitiva en una interfaz gráfica.

4.1. Requerimientos

El requerimiento para este sistema son de tipo hardware y software en el cual estos son de gran importancia para la ejecución de nuestro proyecto y se encuentran detallados a continuación.

4.1.1. Hardware

- Módulo Tarjeta Nvidia Jetson TX2.

  Que permite el procesamiento de imágenes gracias a sus 256 núcleos de GPU.

- Servidor Web con Base de Datos

  Con Sistema Operativo Ubuntu Server para levantar servicios de apache y MySql.

- Servidor con Tarjeta Gráfica Nvidia

  Con capacidades técnicas de 32Gb de RAM y 128Gb de almacenamiento.

4.1.2. Software

- Nvidia DIGITS.

  Un contenedor para TensorFlow que proporciona una interfaz web gráfica para entrenar rápidamente redes neuronales profundas (DNN) de alta precisión.

- CUDA.

  Controladores de la tarjeta gráfica de Nvidia (Drivers).

- Servicios Linux: SSH y SFTP

  Son nombres de protocolos y de los softwares que los implementa para el acceso remoto de servidores que garantizan el intercambio y transferencia de datos a través de un canal seguro.

4.2. Requerimientos del Aplicativo

4.2.1. Requerimientos Funcionales

Código	RF01
Nombre	Autenticación de Usuario
Descripción	El sistema debe tener un proceso de inicio de sesión que es de carácter obligatorio, para tener el acceso a las funcionalidades del sistema.
Características	Necesita el usuario y la contraseña para poder iniciar sesión
Prioridad	ALTA

Tabla 4.1: Requisito Funcional 1

Código	RF02
Nombre	Registro de Persona Desaparecida
Descripción	El usuario debe ingresar todos los parámetros o campos de la persona desaparecida.
Características	Todos los campos son de carácter obligatorio, si no ingresa un parámetro no podrá guardar la información de la persona.
Prioridad	ALTA

Tabla 4.2: Requisito Funcional 2

Código	RF03
Nombre	Información de la Persona Desaparecida
Descripción	Muestra toda la información de la Persona.
Características	Muestra la información de la persona lo más importante que podemos visualizar de la imagen.
Prioridad	MEDIA

Tabla 4.3: Requisito Funcional 3

Código	RF04
Nombre	Consumo de Api Rest
Descripción	Conexión del aplicativo web con la plataforma de DIGITS.
Características	Aquí realizamos la conexión con el DIGITS, y con ello logramos la creación del conjunto de datos y el modelo para el entrenamiento de nuestra red neuronal profunda.
Prioridad	ALTA

Tabla 4.4: Requisito Funcional 4

Código	RF05
Nombre	Transferencia de Archivos
Descripción	Transferir los archivos al superordenador Nvidia Jetson TX2.
Características	Realizamos él envió de los archivos hacia la Jetson, con el que nos ayudara para la ejecución de nuestro modelo en la cámara en tiempo real.
Prioridad	ALTA

Tabla 4.5: Requisito Funcional 5

Código	RF06
Nombre	Encender la Cámara
Descripción	Encender la cámara del superordenador Nvidia Jetson TX2 para enfocarnos en el despliegue de nuestro modelo y tener los resultados en tiempo real.
Características	Conexión y ejecución del modelo con el super-ordenador Nvidia, con esto obtenemos los resultados en tiempo real como es la predicción de la persona.
Prioridad	ALTA

Tabla 4.6: Requisito Funcional 6

Código	RF07
Nombre	Notificaciones
Descripción	Envía notificaciones de la búsqueda de la persona, por ejemplo, la persona Andrés Garnica fueencontrada en el sector del aeropuerto.
Características	Enviar datos de predicción desde el superorde-nador hacia el aplicativo web a través de librerías de python
Prioridad	ALTA

Tabla 4.7: Requisito Funcional 7

4.2.2. Requerimientos No Funcionales

Código	RNF01
Nombre	Interacción del Usuario con el Software
Descripción	Software amigable hacia el usuario.
Características	Interfaz intuitiva, con la inclusión de la librería sweet alert
Prioridad	MEDIA

Tabla 4.8: Requisito No Funcional 1

Código	RNF02
Nombre	Integridad de datos
Descripción	Cifrar la información.
Características	Garantizar la integridad de la información como en la seguridad a través de SFTP.
Prioridad	ALTA

Tabla 4.9: Requisito No Funcional 2

Código	RNF03
Nombre	Selección de Sistema Operativo
Descripción	Analizar a detalle de todas las especificaciones y requisitos para tener un buen rendimiento en la ejecución del proyecto.
Características	Ver todos los detalles del software a utilizar como su rendimiento, almacenamiento entre otros.
Prioridad	ALTA

Tabla 4.10: Requisito No Funcional 3

Código	RNF04
Nombre	Recursos de la Plataforma
Descripción	Indagar todos los detalles y especificaciones de la plataforma a utilizar.
Características	Almacenamiento, rendimiento
Prioridad	ALTA

Tabla 4.11: Requisito No Funcional 4

4.3. Diagrama para la aplicación Web

4.3.1. Diagrama Entidad-Relación

El Diagrama Entidad-Relación es de vital importancia para poder identificar, observar el comportamiento de las clases que están involucradas en el sistema.

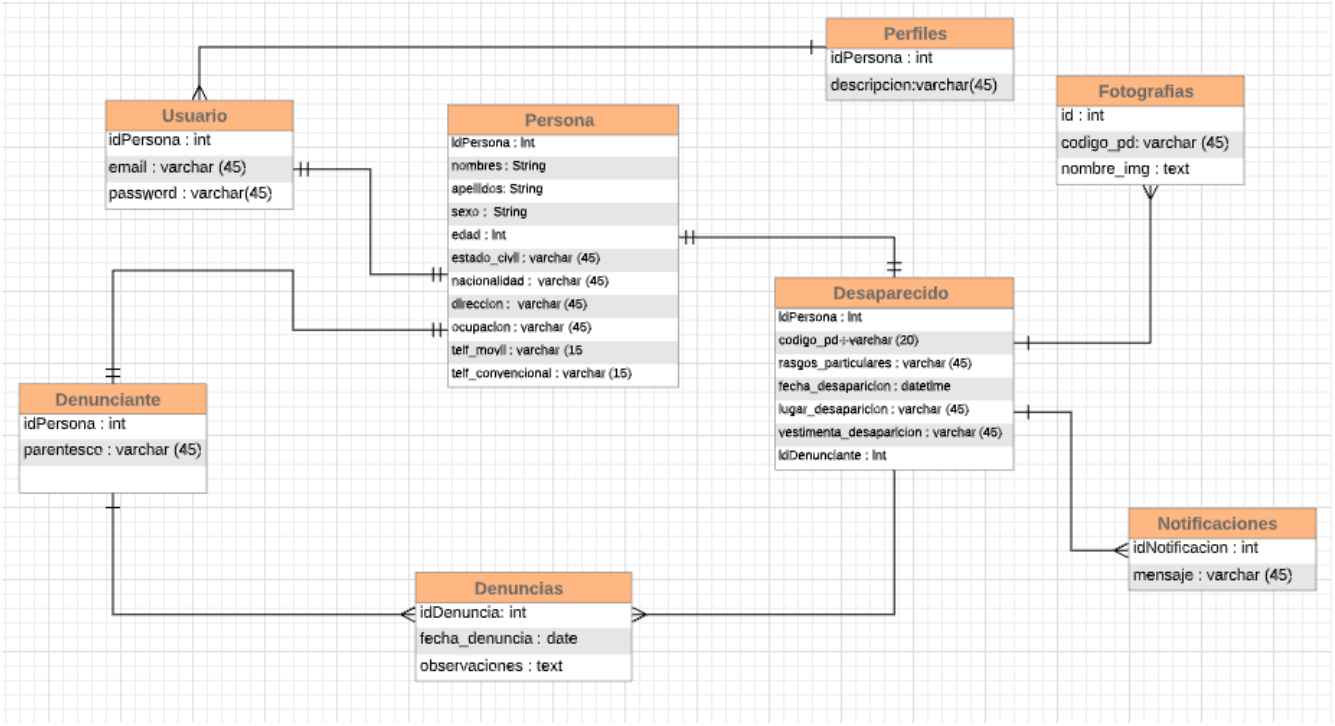


Figura 4.2: Diagrama Entidad Relación

Como se observa en la Figura 3.2, el sistema tiene varias tablas pero las que destacarán a nivel de interfaz gráfica son:

La tabla Usuario: Se encarga de almacenar toda la información tanto de la persona desparecida como de la denunciante.

La tabla Denunciante: Como su nombre lo indica se encarga de proporcionar la información de la persona desparecida.

La tabla Denuncias: Esta es encargada de guardar la fecha de la denuncia y también proporciona información de cómo fue la desaparición de la persona.

4.3.2. Diagrama Casos de Uso

El Diagrama Casos de Uso son elementales para ver la interacción o funciones del sistema o aplicativo tanto con el usuario final u otros sistemas, a continuación iremos observando varios diagramas.

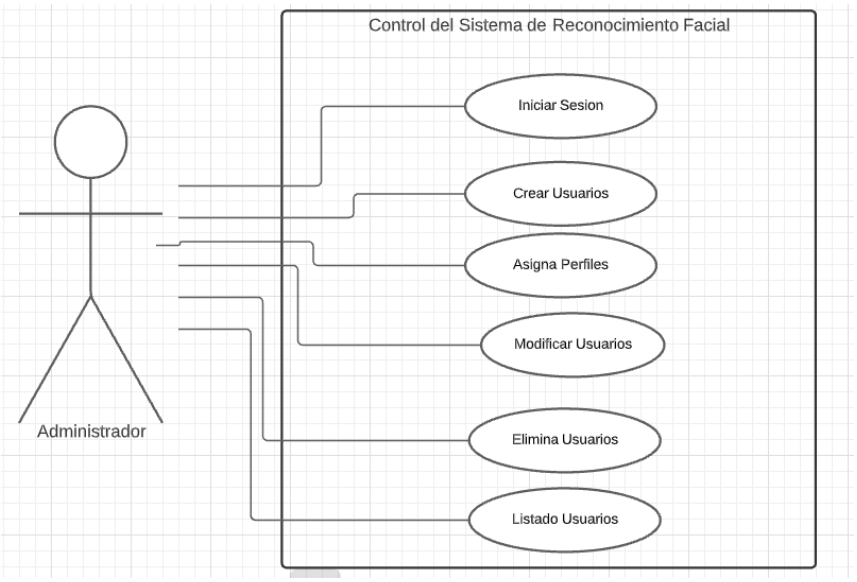


Figura 4.3: Diagrama Caso de Usos del Administrador

Como podemos observar en la figura 3.2 es la interacción del administrador en el aplicativo, en el cual va ser el encargado de crear, modificar y eliminar.

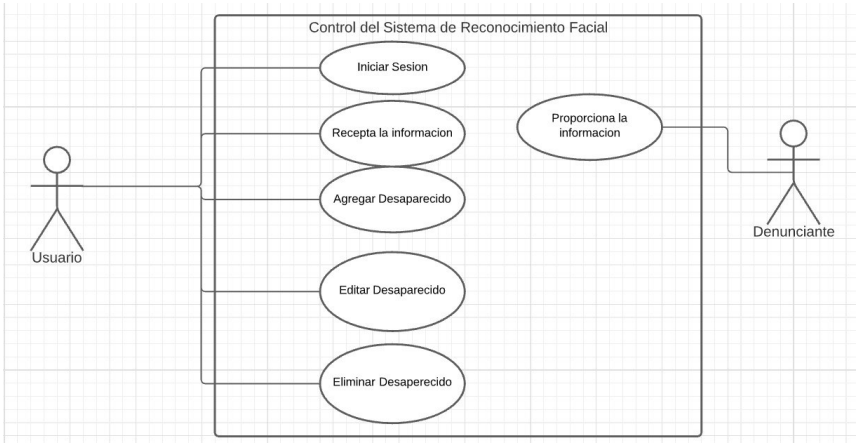


Figura 4.4: Diagrama Caso de Usos del Usuario Denunciante

Usuario Denunciante como podemos observar en la figura 3.3.

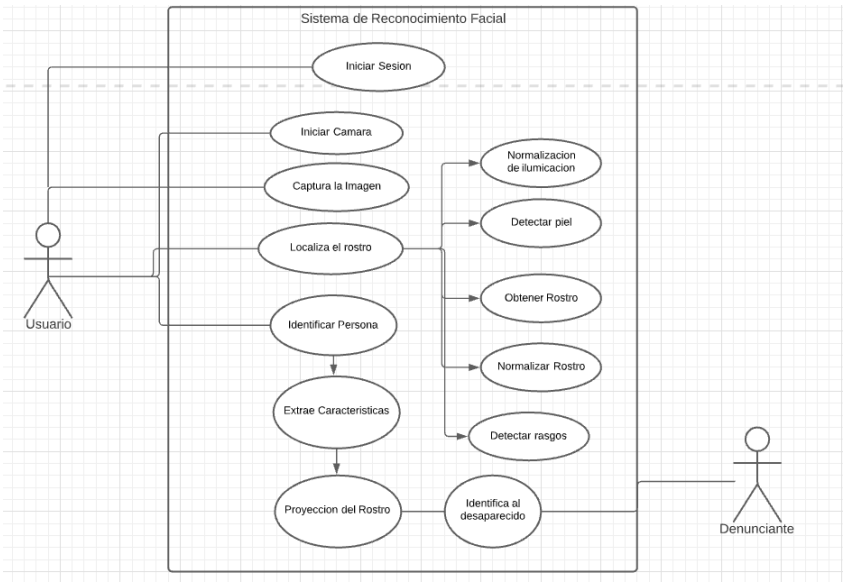


Figura 4.5: Diagrama Caso de Usos Sistemas de Reconocimiento Facial

4.3.3. Diagrama de Secuencias

En el primer Diagrama de Secuencia podemos observar el proceso de como el administrador inicia sesión y posteriormente realiza el CRUD del aplicativo por último se puede ver el listado general.

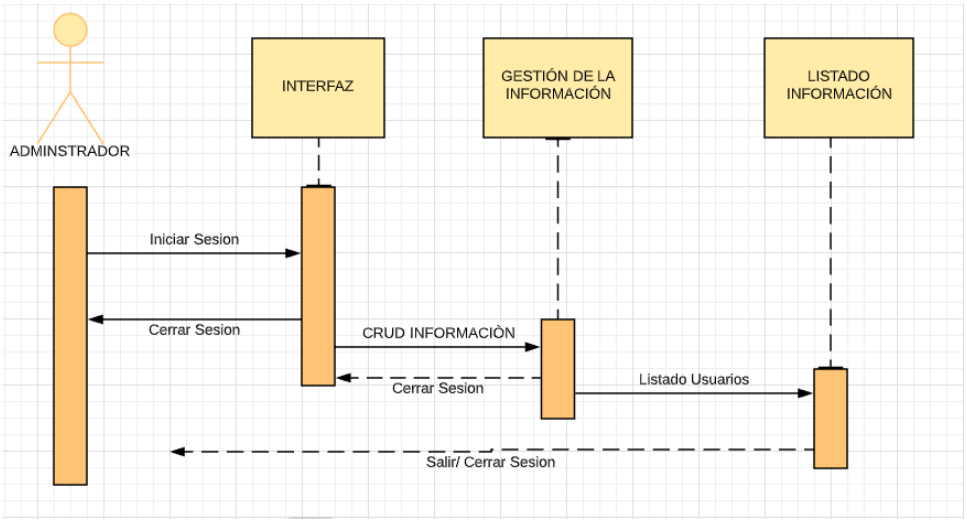


Figura 4.6: Diagrama de Secuencia de la Aplicación WEB

En el segundo Diagrama de Secuencia podemos observar el proceso de como el usuario inicia sesión posteriormente recepta toda la información o datos de la persona desaparecida al denunciante por último lista toda la información otorgada para proceder a la búsqueda.

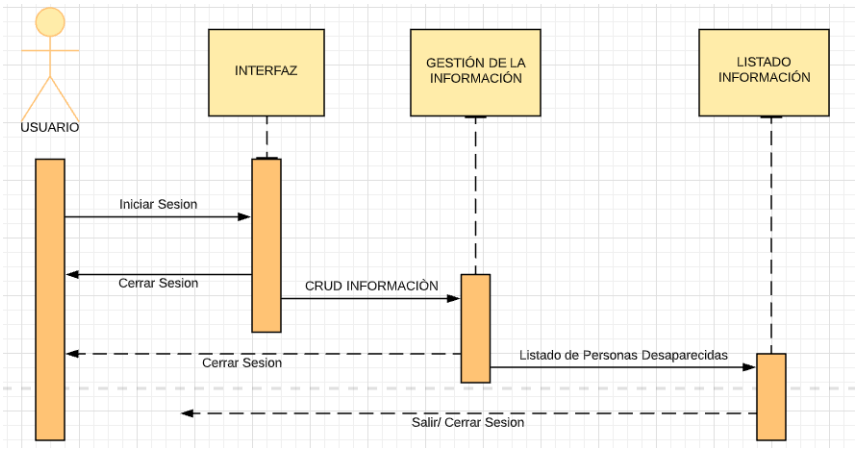


Figura 4.7: Diagrama de Secuencia de la Aplicación WEB

En el último Diagrama de Secuencia podemos observar el proceso es el mismo inicia sesión y con ello el aplicativo se empieza a ejecutar en el cual se enciende la cámara y empieza a realizar la búsqueda de la persona empieza a verificar las características del entrenamiento y son las correctas detecta la persona.

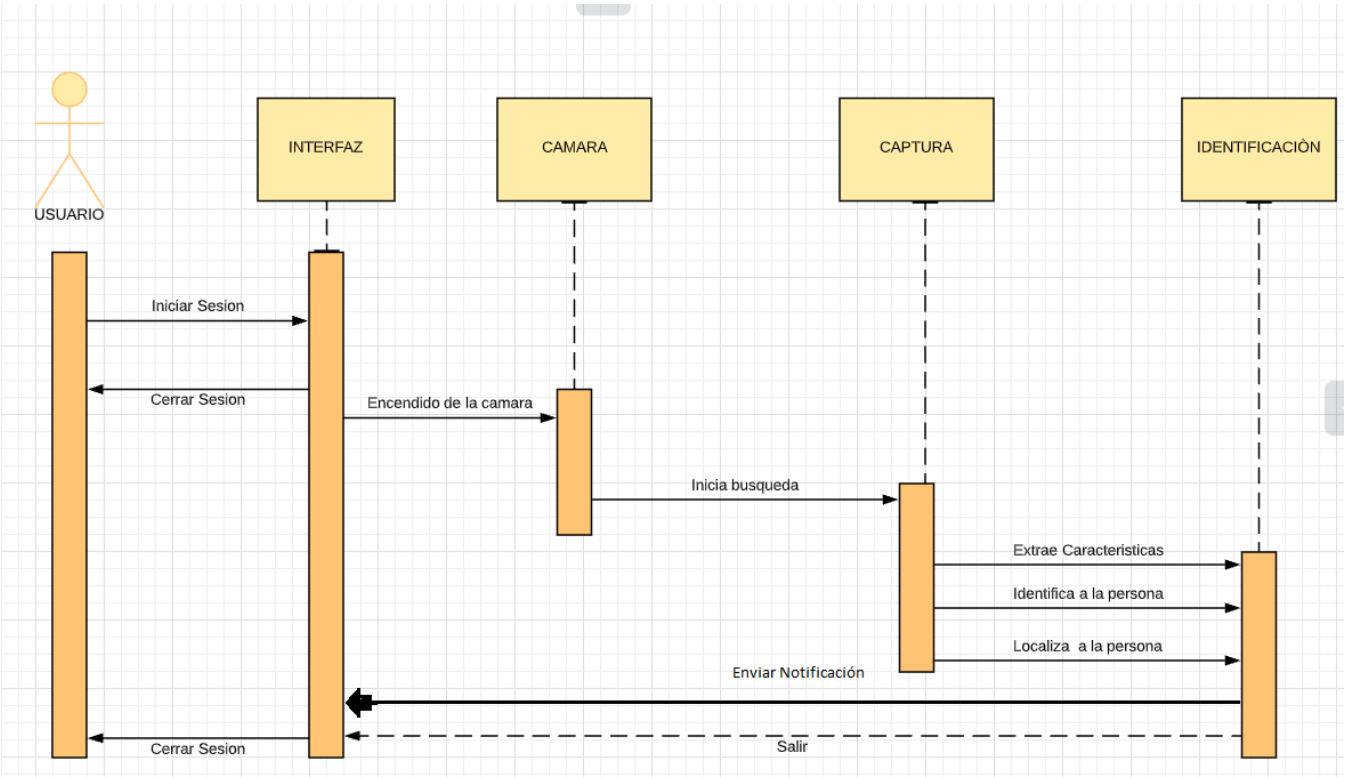


Figura 4.8: Diagrama de Secuencia de la Aplicación WEB

4.4. Desarrollo

A continuación, vamos a describir por etapas todo el procedimiento de nuestro proyecto en donde se explicará acerca de la creación de la red neuronal especificando todos sus parámetros para el conjunto de datos y el modelo. Posteriormente trataremos sobre la etapa de desarrollo del aplicativo web en donde abordaremos sobre el gestor de datos y también explicaremos sobre los servicios que se desplegaron para la comunicación con la plataforma de ***DIGITS***. Por último, trataremos sobre la configuración de la tarjeta Nvidia Jetson TX2, en donde veremos sobre los requisitos que necesitamos para la misma y también para la ejecución de la cámara con su respectivo modelo.

4.4.1. Etapa Creación DNN(Red Neuronal Profunda)

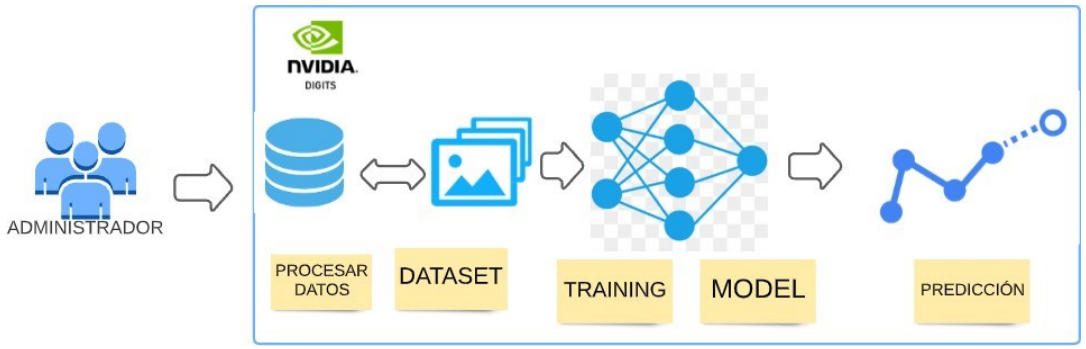


Figura 4.9: Etapa Creación DNN

En la actualidad las redes neuronales profundas (por sus siglas en ingles DNN) no es más que la cantidad de capas a procesar, por lo tanto esta se encarga de recibir un conjunto de datos donde estos internamente se procesan, analiza y realizan cálculo, al terminar ese procedimiento interno se obtiene una salida es decir los resultados de dicha red. En resumen esta red DNN se utilizó porque estas aprenden acciones repetitivas como es el de nuestro caso en la clasificación de imágenes dado que este procedimiento se realizará para la creación de la red neuronal profunda.[23]

Con respecto, a la creación de la DDN o Red Neuronal Profunda en la plataforma de Nvidia denominada ***DIGITS*** debemos seguir el siguiente procedimiento como observemos en la figura 4.9, en el cual en primera instancia debemos procesar la información o el conjunto de datos, para ello, cabe resaltar que la estructura de las imágenes es de gran importancia para el entrenamiento de la red estas deben estar por subdirectorios y que cada de uno ellas deben tener de 1 o más imágenes mientras más cantidad imágenes la predicción será más precisa.

Asimismo, una vez indicado la estructura de las imágenes en el punto anterior, empezamos con la creación del dataset en la que primero debemos iniciar sesión, posteriormente nos aparecerá una ventana en donde tenemos que ir llenando los siguientes campos que a continuación

iremos detallando [24].

- **Tipo de Imagen:** El tipo de imagen en la que va a trabajar ya sea a color o gris
- **Tamaño de Imagen:** El tamaño de la imagen en el que va a trabajar para el entrenamiento.
- **Transformación de Dimensionamiento:** El método de transformación de imágenes al cambiar la resolución por ejemplo crop, el cual es de recortar.
- **Entrenamiento de Imágenes:** El directorio en donde se encuentra las imágenes.
- **Porcentaje para Validación:** EL porcentaje de validación en el que se trabaja para el entrenamiento el recomendado es de 10 % o 25 %.
- **Porcentaje para Pruebas:** El porcentaje del testeo se recomienda entre el 3 % -5 %
- **Codificación de Imagen:** Este indica el formato de archivo en la que se va a trabajar. Nosotros nos enfocamos en png
- **Nombre del Grupo:** Este parámetro es el grupo donde va a pertenecer el dataset
- **Nombre del Conjunto:** Este parámetro es el nombre que le vamos a asignar a nuestro dataset.

Una vez llenado todos los campos procedemos a crear el dataset, el cual se demora dependiendo de la información a procesar.

En cuanto a la creación del **Model** o Modelo, debemos seguir el mismo procedimiento de la figura 3.9 en el que nos aparecerá una interfaz para la creación del mismo, por otra parte, debe estar creado el dataset sino esta no procederá a generar el modelo. A continuación veremos las opciones más influyentes para la creación del modelo.

- **Seleccionar conjunto de datos:** Aquí seleccionamos el dataset creado en el paso anterior.

Parámetros Recomendados:

- **Entrenamiento de Épocas:** Numero de épocas o iteraciones, esta se refiere a las capas a procesar en el que nosotros colocaremos el valor de 30.
- **Épocas de Instancia:** Este es un intervalo de épocas, en el cual se va creando un **snapshot** de la red neuronal este depende mucho de las épocas en la que está trabajando.
- **Intervalo de Validación:** Este parámetro depende de las épocas en las que se está trabajando. Se recomienda trabajar en un valor de 10 %.
- **Tipo de Solucionador:** Este el algoritmo de entrenamiento, en el cual nosotros trabajamos en el SGD.

- **Tasa de Aprendizaje Base:** Es el margen de error durante el entrenamiento La tasa de aprendizaje en el cual nosotros trabajamos con el valor de 0.001.
- **Redes Estandarizadas:** Aquí usaremos una red pre-entrenada en nuestro caso elegiremos AlexNet para nuestro entrenamiento.
- **Nombre del Grupo:** Este parámetro es al grupo al que va a pertenecer el modelo.
- **Nombre del Modelo:** Se asigna el nombre de nuestro modelo creado.

Asimismo, llenado todos los campos damos clic en el botón de create y empezará el entrenamiento, este se demora dependiendo de las características del modelo a entrenar. Por último,una vez ya finalizado el entrenamiento de nuestra red, se debe realizar pruebas de testeo, en laque nosotros obtuvimos un 95 % en las predicciones.

4.4.2. Etapa de Desarrollo Web

4.4.2.0.1 Sistema Operativo Linux - Ubuntu

Hoy en día, la mayoría de empresas que trabajan o implementan áreas tecnológicas, lo realizan en un entorno de desarrollo con sistema operativo Linux; y el distro de mayor preferenciapor todo tipo de usuario es Ubuntu, dado que es un sistema operativo ligero y bastante intuitivopara los usuarios finales. La universidad Técnica de Ambato, publicó un artículo relacionado ala importancia de aprendizaje de Ubuntu, cuyo autor detalla que Ubuntu es un sistema operativoestable con soporte y actualizaciones constantes cada cierto tiempo, además de que sus múltiples paquetes de software poseen una licencia libre o de código abierto lo que lo hace ideal parael desarrollo y despliegue de aplicaciones o servicios web o para otros propósitos socialmente útiles convirtiéndole en un sistema operativo rápido y eficiente para usuarios novatos o hasta denivel avanzado [25].

En nuestro caso, se decidió usar Ubuntu para desplegar y testear varios servicios a la vez, para no tener problemas con consumo de recursos por cada servicio y más aún si estos trabajan en paralelo. Servicios de transferencia de archivos (SFTP) y clasificación y procesamiento de imágenes (Nvidia DIGITS) fueron unos de los procesos en las que se trabajó en paralelo, pese a que se tenía un servidor con recursos suficientes el consumo de recursos fue mínimo. Por otra parte, en relación al desarrollo web se desplegó un segundo servidor con apache y una base de datos con gestor MySql, ambos sistemas de código abierto hasta un cierto punto; ya que al menos MySql es un sistema desarrollado bajo una licencia dual, pero que pese a ello ambas permiten la protección de integridad del código, seguridad, flexibilidad, soporte documentado y costos bajos.

Mencionado lo anteriormente dicho, Ubuntu demuestra que es bastante eficaz y rápida al testear aplicaciones o servicios que requieren una carga alta de procesamiento esencialmente, además de que por defecto viene instalado librerías y complementos de Python, un lenguaje de programación considerando como el rey de los lenguajes para proyectos relacionados con inteligencia artificial y procesamiento de imágenes; pero de ello se profundizará más adelante. De igual forma, parte de nuestro proyecto involucra la utilización de un superordenador basado

enIA , Jetson TX2, que está diseñada para soportar cargas altas de trabajo con cálculos bastantes complejos por medio decir un kit de desarrollo implementadas en Ubuntu; teniendo así, una topología bastante estandarizada y compatible.

4.4.2.0.2 Protocolo de Transferencia de Archivos Seguro - SFTP

Linux ofrece múltiples servicios para que sean instalados según la necesidad del usuario. Comúnmente entre los más destacados se encuentran: Apache (web), Conexión remota (ssh), y transferencia de archivos (ftp), servicios que fueron utilizados para el presente proyecto.

Por otra parte, SFTP - Secure File Transfer Protocol sea quizás el servicio más´ importante, pues fue el servicio que permitió la comunicación y transferencias de archivos entre el Servidor con Nvidia DIGITS, la aplicación web y la Jetson TX2, garantizando la integridad y la segundad de la misma. Aunque los servicios de transferencia de archivos (FTP) y conexión remota (SSH) son servicios totalmente independientes, SFTP puede ser considerado como el sucesor de FTP ya que implementa la seguridad de cifrado de SSH, al momento de la transferencia de ficheros.

4.4.2.0.3 Lenguaje de Programación

Al momento de elegir un lenguaje de programación, considerar todo tipo de características y requerimientos es primordial para el desarrollo de un aplicativo web y así garantizar la calidad y el funcionamiento adecuado del software previo al momento de ser utilizado por el usuario. Para este caso se eligió PHP para el despliegue del aplicativo web y Python como complemento para el aprendizaje profundo, reconocimientos de objetos y cálculos de procesamiento complejos que son utilizados por cámara Jetson TX2.

- PHP (Hypertext Pre-Processor)

Entre los lenguajes más utilizados para desarrollo web, PHP fue en uno de los primeros lenguajes de programación que permitía crear páginas web dinámicas revolucionando desde entonces la programación web por backend. Aunque PHP surgió desde los años 90’s, al día de hoy se mantiene entre los más utilizados para desarrollo web por facilidad de aprendizaje. Yolanda Sierra [26] reafirma que, a pesar de que PHP es un lenguaje bastante antiguo posee una arquitectura bastante eficaz, ya que reduce la carga de manteamiento al separar el controlador y la API de alto nivel en sus propias extensiones y librerías. Considerando lo dicho anteriormente, PHP facilita el desarrollo del aplicativo web dado que no tiene problemas con trabajar o migrar a frameworks que incluso agilizarían mucho más sus respectivos procesos.

Ventajas

- Facilidad de Aprendizaje.
- Soporte en casi toda plataforma de alojamiento web
- Combinación de trabajar con código HTML

- Documentación Amplia y de código Abierto
- Aplicaciones desarrolladas con PHP son compatibles con diversos sistemas operativos

Desventajas

- No idóneo para aplicaciones gráficas
- Para testeo requiere un servidor con soporte en PHP
- Al ser lenguaje interpretado puede ser lento a la ejecución de un script de php

- Python

Casi todo usuario prefiere un software personalizado o con un propósito especifico que cumpla sus expectativas y a su vez que sean eficientes en cuestión a tiempo. Para este caso, siendo un proyecto enfocado al procesamiento de imágenes en paralelo donde profundiza bastante enlo que es inteligencia artificial, Python es el rey de lenguajes donde destaca por mucho en estaárea. Su dominio en la inteligencia artificial radica en éxito y eficiencia de sus algoritmos que en la mayoría de casos incluye fragmentos de códigos que logran emular una inteligencia artificial como tal, posicionándose como lenguaje de alto nivel por sobre todo los demás lenguajesen la misma área como lo son R y C++. En otras palabras, Python al ser lenguaje de alto nivel nos brinda con sus librerías complementarias realizar los cálculos que se requiere para el pro- cesamiento de imágenes en paralelo sin que consuma recursos de hardware de manera excesivadentro de la cámara Jetson,

4.4.3. Etapa Configuración de la Tarjeta Nvidia Jetson TX2

4.4.3.0.1 Prerrequisitos

En párrafos anteriores, se mencionó que Tarjeta Nvidia Jetson TX2 no es una cámara como tal, sino un super mini ordenador diseñada para actividades de procesamiento en paralelo; y como todo ordenador tiene instalado por defecto un sistema operativo, este caso es Ubuntu. Sin embargo, al ser un dispositivo de paquete es necesario instalar ciertos paquetes y dependencias para lograr el razonamiento del modelo con la cámara de la Jetson, de las cuales se encuentran detallas a continuación:

- **Jetpack SDK:** una colección de APIs y librerías utilizadas para soluciones de análisis profundo de vídeos inteligentes (IVA) en las plataformas de Jetson.
- **Librerías para Visión Artificial**
  - **OpenCV:** desarrollada por Intel, es una librería que permite el procesamiento de imágenes y potenciada por el procesamiento de múltiples núcleos que posea el GPU y complementada por Numpy, librería de Python diseñada para el soporte de vectores y matrices, obteniendo así la máxima optimización de proceso al momento de segmentar la información obtenida a partir de una imagen. [27]
  - **Caffe:** un Framework enfocado a la Visión Artificial y al desarrollo de DNNs que, a más de ser bastante robusto es mucho más rápido y liviano que otros frameworks existentes como: Keras, Darknet o las propias librerías de Google y Microsoft. [28]

La página oficial de Nvidia ofrece distintas versiones de Jetpack diseñadas para una versión especifica de Sistema Operativo en Linux. Sin embargo, la más estable con la que se realizaron varias pruebas de testeo, a la fecha de publicación del presente documento, fue la versión 3.3 para Ubuntu 16, además de que igualmente se consideró las versiones de las librerías de visión artificial correspondientes a la versión de dicho sistema operativo.

4.4.3.0.2 Despliegue de Modelo Entrenado

En la Etapa de Creación de DNNs se explicó a detalle la creación de un modelo personalizado utilizando Nvidia Digits que al momento de generar y descargar el modelo se crearon varios ficheros internos necesarios para la clasificación de imágenes, estos archivos son necesarios para la Jetson TX2 ya que estos serán interpretados por la cámara al momento de reconocerlos rostros en tiempo real.

Para la lectura de los ficheros del modelo pre-entrenado se utilizó un código Python ya desarrollado y disponible en la web, pero se le realizaron modificaciones como peticiones POST y pase de parámetros para que se adaptara al propósito de nuestro proyecto, de tal manera que ésta pueda ser ejecutado desde una terminal siguiendo la sintaxis a continuación:

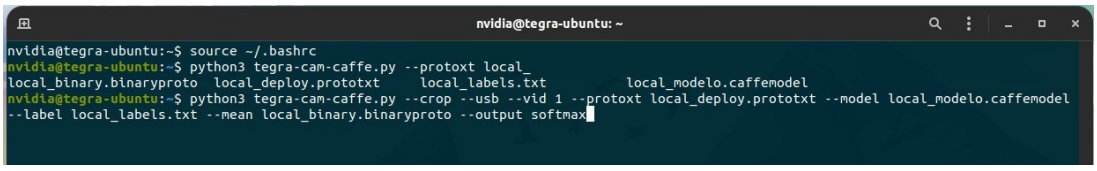


Figura 4.10: Despliegue de DDN con Webcam Jetson TX2

4.4.4. Etapa Despliegue de proyecto

Para el despliegue de nuestro proyecto cabe recalcar que la comunicación e intercambio de datos entre cada servicio es lo primordial y para cumplir con este objetivo se implementó algunos de los servicios linux mencionados anteriormente y una API que ofrece la propia plataformade Nvidia DIGITS para el consumo de servicios.

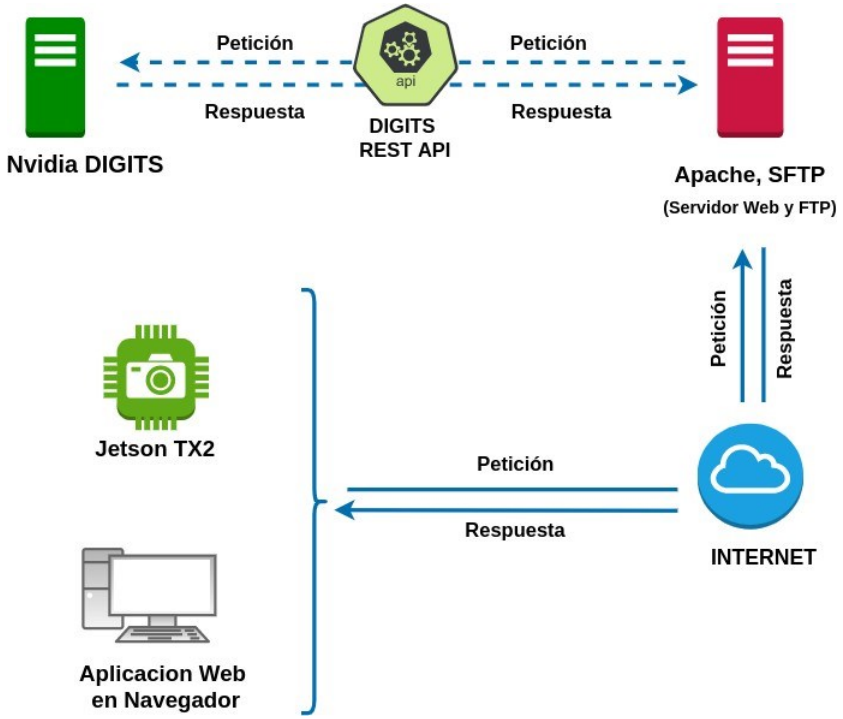


Figura 4.11: Consumo de DIGITS Api Rest

En la figura 3.11, se aprecia el flujo del proyecto, en la que desde un navegador se puede visualizar la aplicación levantada desde el servidor Web, donde el usuario final interactúa únicamente con la aplicación. En este caso el usuario solicita requerimientos al servidor como por ejemplo cargar y almacenar la información del Usuario, donde éste le retornará si se cargó correctamente o sufrió un error durante este proceso. Posteriormente el usuario desde la misma interfaz de la aplicación puede generar el modelo de entrenamiento, pero en esta ocasión el usuario realiza esta petición desde el propio servidor Web hacia otro servidor en la quese encuentra implementado el servicio de Nvidia DIGITS a través de un Api Rest, donde este únicamente solicita la creación del modelo de entrenamiento con un dataset de imágenes que fueron enviadas con la información necesaria desde el servidor web y retornará un mensaje al usuario donde especificara si se creó o no dicho requerimiento.

Previo al consumo del Api Rest de DIGITS, internamente el servidor web solicita una conexión SFTP al Servidor con Nvidia DIGITS donde la conexión sea exitosa, hace el intercambiode datos entre las dos partes, cuya información contiene el dataset de imágenes a clasificar, los

parámetros de entrenamiento de clasificación de imágenes y generación de modelo, todo esto en un formato flexible como lo son: JSON, XML o Texto Plano; y gracias a que el servicio de SFTP incluye el cifrado de SSH, los datos se intercambian de manera segura.

Al momento de que el servidor con DIGITS retorne una respuesta exitosa al servidor Web, el usuario procederá sin problemas a descargar los archivos necesarios del modelo entrenamiento para la ejecución en la cámara de la Jetson TX2. Para ello el servidor Web realiza una nueva conexión SFTP a la Jetson TX2 para proceder a transferirle los archivos requeridos. Una vez finalizado ese proceso el usuario final mandará una petición al Servidor Web para encender cámara y proceda a reconocer un rostro especificado en tiempo real, que al momento que la cámara Jetson TX2 reconozca el rostro de la persona especificada cuyo porcentaje de coincidencia supere el 90 %, retornará a la aplicación una notificación detallando que la persona fue localizada por dicha cámara ubicada en un punto estratégico.

Capítulo V

RESULTADOS

En este capítulo abordaremos sobre los resultados del aplicativo web, tanto en la interacción con la plataforma de digits para la creación de la red neuronal y en el despliegue hacia la tarjeta Nvidia Jetson TX2, en donde se observarán los resultados de la red neuronal a través de la cámara y estos al tener coincidencias precisas enviará notificaciones hacia el aplicativo web.

5.1. Plataforma de Reconocimiento Facial

Como resultado general, se tiene una plataforma de reconocimiento facial, en el cual se emplean modelos para identificar personas, por ende esta plataforma se encarga de crear un modelo para la identificación de personas desparecidas y estos ser desplegados a través de un módulo Nvidia en tiempo real.

5.2. Fase Creación de la Red Neuronal Profunda

Creación de la Red Neuronal

En la creación de la red neuronal a través de la plataforma digits el conjunto de datos es de gran importancia debido a que este crea 2 bases de datos tanto para entrenamiento como para validación, dado que estos contienen la información o datos de las imágenes clasificadas como observa en la figura 4.1 y 4.2.

Figura 5.1: Dataset DB train

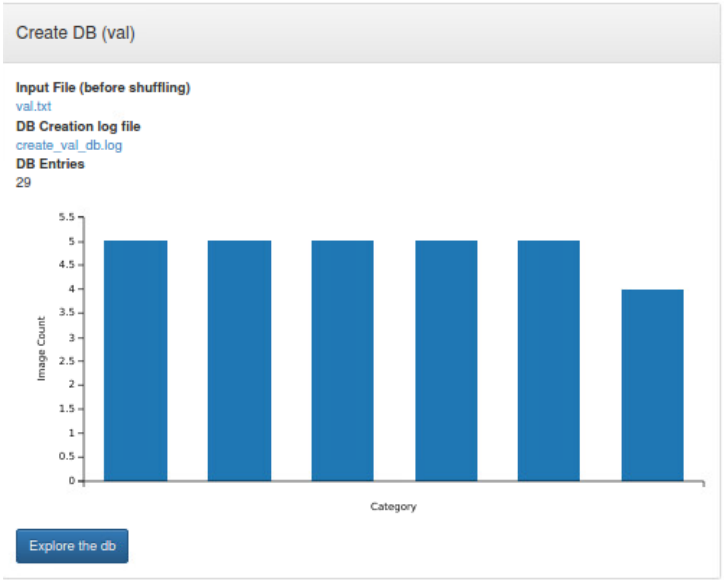


Figura 5.2: Dataset DB val

En esta sección vamos analizar resultados acerca de la predicción y pérdidas durante la clasificación de imágenes que se visualiza en la figura 4.3 en esta obtuvimos un porcentaje alto en el entrenamiento debido a que las imágenes previamente cargadas cumplían con requisitos específicos.

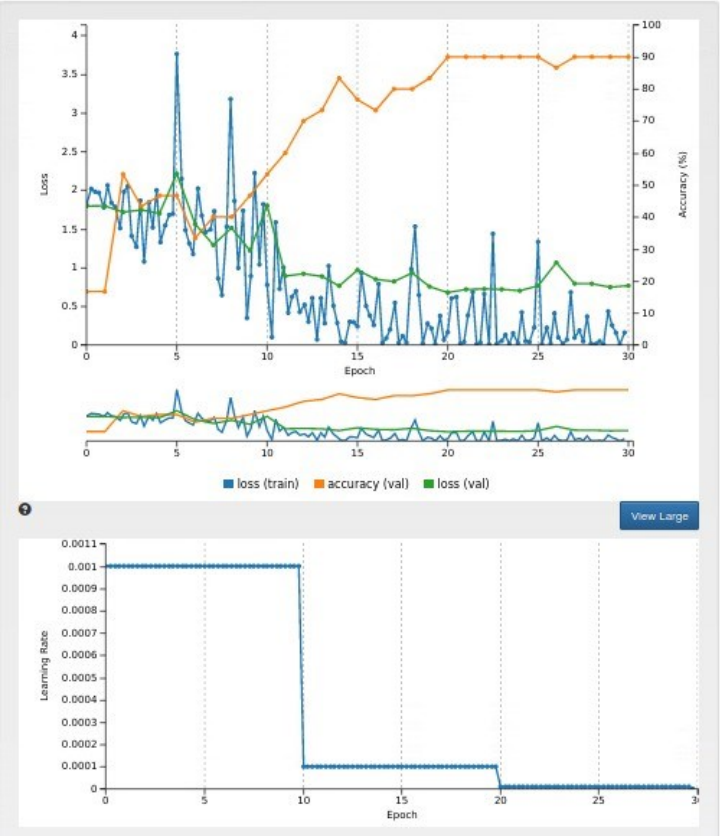


Figura 5.3: Resultados de Predicción y perdida

Por último, vamos a realizar una prueba en la plataforma de digits en la seleccionamos una imagen de entrada y con ello vamos a observar los resultados del entrenamiento en este caso la predicción ha sido del 99.99 %, por lo que con este resultado nos ayuda para poder identificar ala persona con mayor precisión.

Figura 5.4: Resultados de Predicción

También es importante resaltar el trabajo de los epochs o épocas en el que podemos evidenciar que analiza todos los rasgos de la imagen así que esa es su función en el que analizan paso a paso la imagen como se observa la figura 4.5 y esta nos ayuda para poder identificar a la persona.

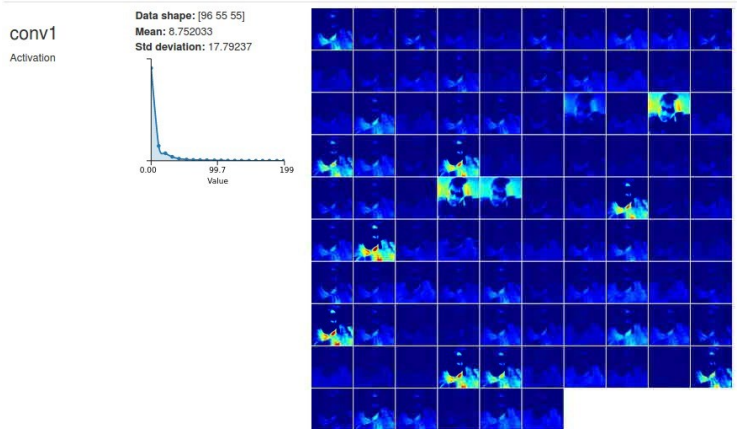


Figura 5.5: Épocas

Con este resultado, cumplimos el objetivo de entrenar validar modelos para la detección de rostros de personas desaparecidas mediante redes neuronales profundas de aprendizaje como se explicó anteriormente.

5.3. Gestión y Administración

Se implemento una plataforma web para gestión y administración de personas desaparecidas en donde aquí insertamos todos sus datos pero el de mas importante es la fotografía el cualnos sirve para realizar el entrenamiento de nuestro modelo.

Figura 5.6: Administración Plataforma

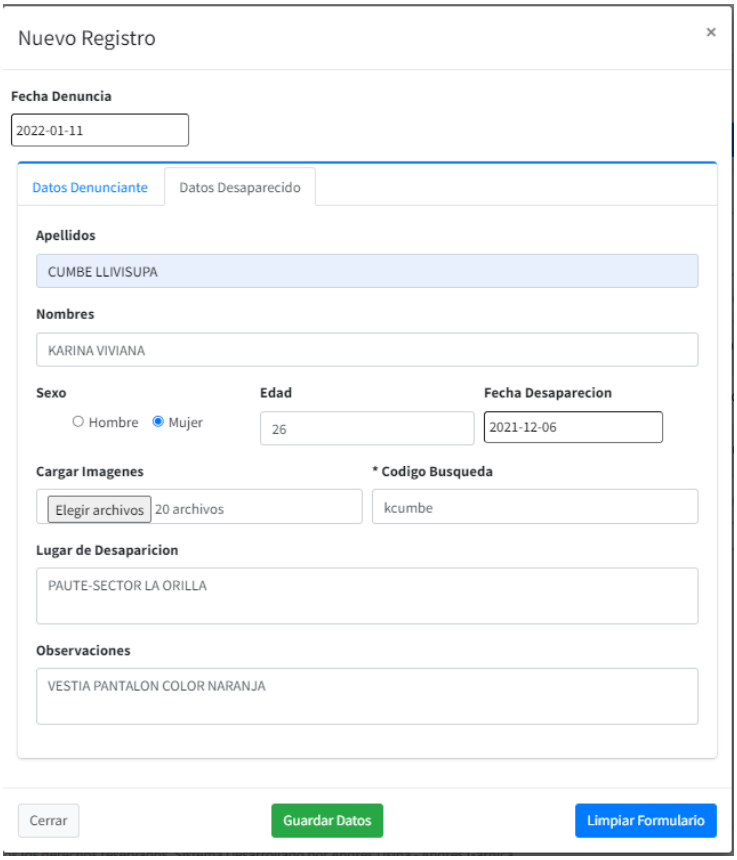


Figura 5.7: Insertar Desparecido

Una vez insertado los datos, podemos observar todas las personas registradas en una tabla y estas serán previas para el entrenamiento de nuestro modelo como veremos a continuación en la figura 4.7.

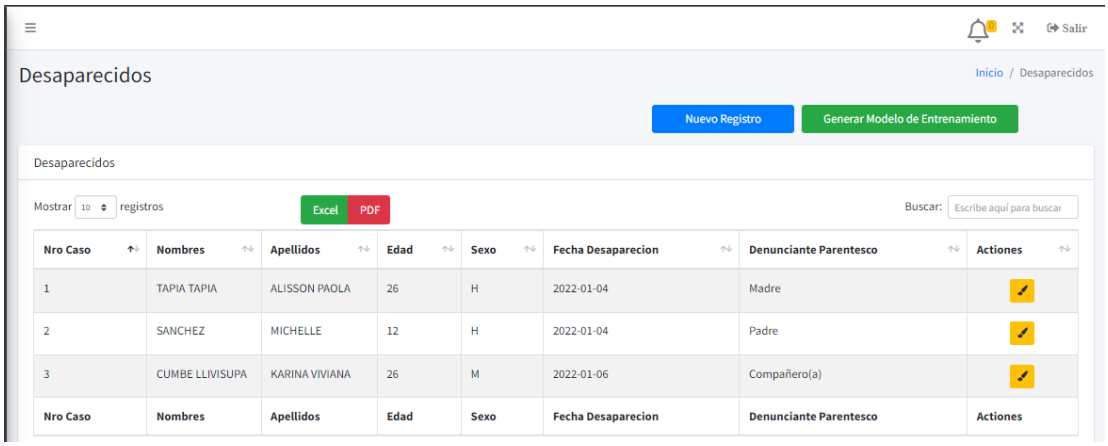


Figura 5.8: Lista de Desparecidos

Con esto cumplimos nuestro 4 objetivo específico en su totalidad, como se observa en las figuras antes mencionadas.

5.4. Despliegue

En esta sección se detallará principalmente los dos puntos clave que permitieron desplegar el proyecto que son: la generación del DDN y la transferencia de archivos entre dispositivos.

La creación de la red neuronal profunda consta de dos fases: la creación del dataset de imágenes y la generación del modelo de entrenamiento; para realizar estos procesos desde la aplicación web se implementó el Api Rest de la plataforma DIGITS en lenguaje de programación PHP, enviando los parámetros requeridos en texto plano para que sea ejecutados desde elcomando *cURL*, tal como se muestra en la figura 4.9.

```
  shell_exec("curl ".$srv_digits.":5000/login -c digits.cookie -XPOST -F username=andrew");

$creaDS ="curl ".$srv_digits.":5000/datasets/images/classification/json
-b digits.cookie -XPOST -F folder_train=".$rutaEntranamiento." -F encoding=png
 -F resize_channels=3 -F resize_width=".$anchoImg." -F resize_height=".$altoImg."
  -F batch_size=".$batch_size." -F solver_type=".$solver_type." -F learning_rate=".$learning_rate."
  -F method=folder -F group_name=".$nombreGrupoDS." -F dataset_name=".$nombre_dataset;

$status_DS = shell_exec($creaDS." 2>&1; echo $?");
```

Figura 5.9: Creación de Dataset por medio de Api Rest

Una vez finalizado la creación del dataset, se procede a generar el modelo de entrenamiento. Tal como se ve en la figura 4.10, la manera de ejecutarlo es bastante similar al proceso anterior, pero en esta fase se debe tener claro los parámetros específicos que se enviarán; como

sonlas épocas, tamaño de lote y el de gran importancia que el Identificador Único (ID) del conjunto de datos creado en la fase anterior, porque sin ello el modelo de entrenamiento no se generará.

```
$creaModeloPreEntrado = "curl ".$srv_digits.":5000/models/images/classification/json -b digits.cookie
-XPOST -F method=standard -F standard_networks=".$red_estandarizada." -F train_epochs=".$cant_epocas."
 -F batch_size=".$batch_size." -F solver_type=".$solver_type." -F learning_rate=".$learning_rate."
 -F framework=caffe -F model_name=".$nombre_modelo." -F group_name=".$nombreGrupoMDL." -F dataset=".$id_ultimo_DS."";

$status_Model = shell_exec($creaModeloPreEntrado." 2>&1; echo $?");
```

Figura 5.10: Creación de modelo por medio de Api Rest

Por otro lado, para transferir los archivos del servidor Web al módulo Jetson TX2, se utilizó librerías ftp que ofrece PHP; en la figura 4.11 se muestra un fragmento de código que permite la conexión entre ambas partes, cuyo propósito es únicamente transferir archivos si la conexión fue exitosa.

```
if($generaDeploy && $generaCaffe && $generaBinary && $generaLabels){
    $checkArchivosDescargados = "ok";
    if($checkArchivosDescargados == "ok"){
        $ftp_svr ="172.16.26.34";
        $ftp_usuario = "nvidia";
        $ftp_pass ="nvidia";
        $ftp_conn = ftp_connect($ftp_svr);
        $login = ftp_login($ftp_conn,$ftp_usuario, $ftp_pass);

        if(!$ftp_conn || (!$login)){
            echo "no se pudo conectar";
            exit;
        }else{
```

Figura 5.11: Conexión FTP en PHP

El resultado que se observa en la figura 4.12 tiene una predicción del 95 % debido que este tiene rasgo fundamental que son los lentes, pero si no es tuviera con la mascarilla la predicción fuera más elevada.

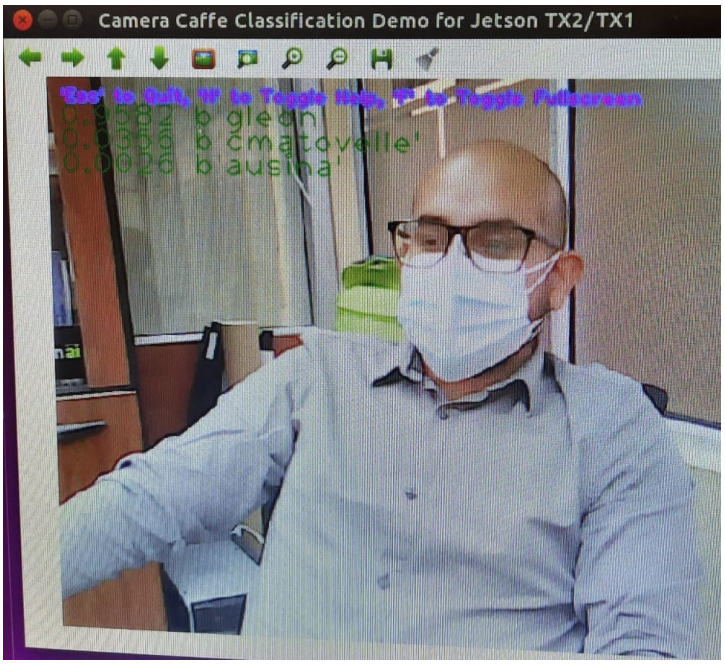


Figura 5.12: Reconocimiento Facial en tiempo real

5.5. Pruebas y Funcionamiento

Las pruebas realizadas sobre Nvidia DIGITS principalmente consistieron en entrenar al modelo periódicamente con múltiples imágenes de distintas personas en varios formatos, resoluciones, tamaño, colores, objeto de imagen, etc.; las cuales fueron mejorando el porcentaje de predicción y a su vez obteniendo resultados constructivos acerca de cómo se puede mejorar el modelo.

Primero se analizó los parámetros necesarios tomadas de la documentación e información de otros proyectos similares realizadas por otros autores como se mencionan en el capítulo 2 en la sección 2.8 de Trabajos Relacionados y posteriormente se procedió a generar el modelo con 2 categorías a clasificar con 15 fotografías cada una, para probar y determinar su funcionalidad y configuración de parámetros.

Como se mencionó en el párrafo anterior se utilizado distintas fuentes de datos de diferentes tipos y conforme se aumentaba la cantidad de datos se fueron determinando los requisitos que debían cumplir las imágenes para la generación del modelo, dando como resultado una predicción de casi el 100 % tal como se visualiza en la Figura 4.4

En la figura 4.6 se muestra una interfaz de usuario desarrollado para este proyecto, la cual se conecta con los otros servidores por medio del API de DIGITS y los servicios SFTP y SSH que se explicaron en el capítulo 3. Resumiendo el funcionamiento completo del aplicativo web se segmenta en 3 secciones: la sección de registro de información del usuario que carga los datos y un conjunto de imágenes en el servidor web, una sección donde estos datos son enviados al servidor con DIGITS a través de una API para el entrenamiento de modelo, y la sección de

despliegue del modelo entrenado en la cámara de la Jetson TX2 que lo que hace es transferir los archivos generados por el servidor DIGITS a la Jetson a través del servicio SFTP; visualizados en las figuras 4.13 y 4.14 respectivamente. Para concluir, en la figura 4.14 se aprecia el consumo de recursos de la aplicación web y su interacción entre el Servidor DIGITS y Jetson, demostrando que el implementar una aplicación web e interactuar con otros dispositivos desdela misma interfaz para reconocer rostros en tiempo real ofrece un rendimiento adecuado.

```
reconocimiento-facial@reconocimientofacial-virtual-machine:~$ ls -l /var/www/rpd
/files_model/
total 223056
-rw-r--r-- 1 www-data www-data    786446 ene 12 15:30 local_binary.binaryproto
-rw-r--r-- 1 www-data www-data      4667 ene 12 15:30 local_deploy.prototxt
-rw-r--r-- 1 www-data www-data        71 ene 12 15:30 local_labels.txt
-rw-r--r-- 1 www-data www-data 227605855 ene 12 15:30 local_modelo.caffemodel
reconocimiento-facial@reconocimientofacial-virtual-machine:~$
```

Figura 5.13: Archivos generados por DIGITS almacenados en el Servidor Web

```
Last login: Wed Jan 12 17:41:56 2022
nvidia@tegra-ubuntu:~$ ls -l | grep local_
-rw-------  1 nvidia nvidia    786446 ene 12 20:30 local_binary.binaryproto
-rw-------  1 nvidia nvidia      4667 ene 12 20:30 local_deploy.prototxt
-rw-------  1 nvidia nvidia        71 ene 12 20:30 local_labels.txt
-rw-------  1 nvidia nvidia 227605855 ene 12 20:30 local_modelo.caffemodel
nvidia@tegra-ubuntu:~$
```

Figura 5.14: Archivos Transferidos desde DIGITS a Jetson TX2

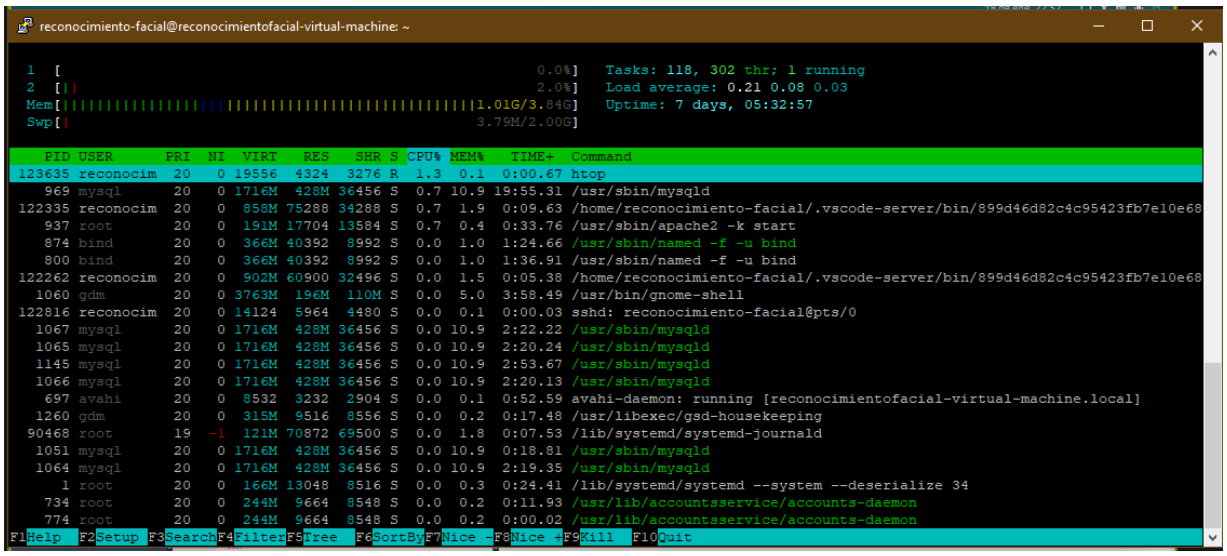


Figura 5.15: Consumo de Recursos del Servidor WEB

CRONOGRAMA

ACTIVIDADES

OE1. Realizar un análisis de las técnicas de visión por computadora utilizadas para la detección de rostros en tiempo real.

- ACT.1.1 Determinar los requerimientos de hardware esenciales utilizados en detección de rostros.
- ACT.1.2 Recopilar información de herramientas de software que operen con cámaras de alta visión.
- ACT.1.3 Investigación de librerías o métodos que permitan la detección de rostros.
- ACT.1.4 Seleccionar un lenguaje de programación compatible con librerías y con el hardware requerido

OE2. Evaluar y Seleccionar los mejores modelos del estado del arte para detección de rostros de personas desaparecidas.

- ACT.2.1 Determinar los requerimientos de software esenciales utilizados para deep learning.
- ACT.2.2 Recopilar información sobre las plataformas para el entrenamiento del deep learning.
- ACT2.3 Investigación de modelos de aprendizaje existentes en la detección de rostros.
- ACT2.4 Investigación de librerías para el entrenamiento de modelos de detección de rostros.
- ACT.2.5 Seleccionar una plataforma compatible con el software, librerías y modelos requeridos para la detección de rostros de personas desaparecidas.

OE3. Entrenar y validar modelos para la detección de rostros de personas desaparecidas mediante redes neuronales profundas de aprendizaje.

- ACT. 3.1 Recopilar información acerca del funcionamiento del reconocimiento facial.

- ACT. 3.2 Dar tratamiento a imágenes con todo tipo de información que ayude a mapear de manera detallada las características de un rostro. Ejemplo: patrones, relieves, muestras, etc.
- ACT. 3.3 Entrenar el algoritmo para el modelo en la detección de rostros.
- ACT 3.4 Aplicar metodología heurística de prueba y error al algoritmo de Reconocimiento Facial con varias imágenes.

OE4. Desarrollar una aplicación para la gestión y administración del sistema de personas desaparecidas.

- ACT.4.1 Levantar requisitos a través de diagramas de casos de uso.
- ACT.4.2 Desarrollar diagrama Entidad – Relación.
- ACT.4.3 Validación de los diagramas Entidad - Relación y Casos de Uso.
- ACT.4.4 Diseñar la Interfaz de Usuario
- ACT. 4.5.Desarrollar aplicación distribuida en un framework de preferencia
- ACT.4.6 Realizar pruebas de funcionamiento en la aplicación (cargar o subir imágenes)

OE5. Desplegar modelos entrenados en equipos de hardware de cómputo de alto rendimiento usando cámaras de alta visión en tiempo real.

- ACT. 5.1 Investigar servicios, funciones o librerías para la comunicación entre la red de entrenamiento deep learning con las cámaras de la jetson TX2.
- ACT. 5.2 Implementar servicios para enviar o desplegar el modelo en las cámaras TX2.
- ACT. 5.3 Visualizar y Comprobar el funcionamiento y el tiempo de respuesta entre las imágenes y la red de entrenamiento.

OE6. Realizar pruebas de funcionamiento y rendimiento de la plataforma de reconocimiento facial.

- ACT. 6.1 Realizar una prueba de detección de rostro conjuntamente con la aplicación web desarrollada.
- ACT. 6.2 Verificar el sistema de alarma en la aplicación al detectar el rostro, de su correcto funcionamiento.
- ACT. 6.3 Documentar el manual de usuario para la administración de la aplicación.

A continuación se observará a detalle el CRONOGRAMA DE ACTIVIDADES que se realizó para este proyecto.

Objetivo	Actividad	Duración	Fecha Inicio	Fecha Fin
OE1.	ACT 1.1	7 días	05/06/2020	12/06/2020
	ACT 1.2	12 días	05/06/2020	17/06/2020
	ACT 1.3	12 días	18/06/2020	30/06/2020
	ACT 1.4	12 días	01/07/2020	12/07/2020
Total		31 días		
OE2.	ACT 2.1	11 días	08/06/2020	19/06/2020
	ACT 2.2	11 días	08/06/2020	19/06/2020
	ACT 2.3	12 días	18/06/2020	30/06/2020
	ACT 2.4	14 días	19/06/2020	03/07/2020
	ACT 2.5	23 días	06/07/2020	29/07/2020
Total		71 días		
OE3.	ACT 3.1	13 días	26/06/2020	09/07/2020
	ACT 3.2	11 días	01/07/2020	11/07/2020
	ACT 3.3	29 días	13/07/2020	12/10/2020
	ACT 3.4	125 días	23/07/2021	28/12/2021
Total		178 días		
OE4.	ACT 4.1	6 días	12/06/2020	18/06/2020
	ACT 4.2	16 días	17/06/2020	03/07/2020
	ACT 4.3	11 días	26/06/2020	07/07/2020
	ACT 4.4	4 días	16/11/2020	20/11/2020
	ACT 4.5	11 días	23/11/2020	04/12/2020
	ACT 4.6	4 días	03/01/2021	07/01/2021
Total		52 días		
OE5.	ACT 5.1	20 días	05/03/2021	25/03/2021
	ACT 5.2	27 días	01/04/2021	27/04/2021
	ACT 5.3	20 días	03/05/2021	23/07/2021
Total		67 días		
OE6.	ACT 6.1	150 días	30/06/2021	30/12/2022
	ACT 6.2	81 días	12/10/2021	03/01/2022
	ACT 6.3	26 días	05/01/2020	31/01/2022
Total		257 días		
Total Días		656 días		
Total Horas		900 horas		

Tabla 5.1: Cronograma de Actividades

PRESUPUESTO

En todo proyecto el presupuesto es esencial dado a que a través de ella se logra precisar elementos y aspectos que se desean conocer o cuantificar con el objetivo de llegar a conclusiones.

Presupuesto del Proyecto					
Tipo Costo	Tipo de Recurso	Tipo de Unidad	Unidades	Precio por Unidad	Costo
Variable	Desarrollador Junior	Horas (Mensual)	100	3	300
Variable	Desarrollador Junior	Horas (Mensual)	100	3	300
Fijo	Nvidia Jetson TX2	Kit Desarrollo	1	600	600
Fijo	Laptop Dell Alta Gama	Caja	1	900	900
Fijo	Laptop Asus Alta Gama	Caja	1	950	950
Fijo	Servidor Alta Gama	Unidad	1	45000	45000

Duración del Proyecto	11 meses
Presupuesto Fijo	$ 47450
Presupuesto Variable	$ 6600
Presupuesto Total	$ 54050

CONCLUSIONES

- Mediante un amplio análisis a profundidad de técnicas y métodos utilizadas en proyectos similares enfocados a visón artificial, fue de vital importancia dado a que a partir de ahí se logró obtener las nociones básicas que comprenden esta área, con lo que permitió entender el funcionamiento adecuado del kit de herramientas que ofrece DIGITS; siendo esta última más ligera en procesamiento por los pocos recursos que esta consume a diferenciade sus adversarios como Keras, Darknet, entre otras.

- En base a una investigación previa, a la fecha de publicación del presente documento y acorde al estado de arte para detección de rostros, existen proyectos enfocados a detectar objetos específicos tales como las plantas en el área de medicina y patrones de diseño en el área de arquitectura, pero específicamente en la rama en reconocimiento de rostros son muy escasos. Sin embargo, al tener limitada información acerca de la detección de rostros se priorizó en seleccionar puntos claves (tamaño y resolución de imagen, tipo de objeto y capas de colores), puntos que posteriormente serian de gran importancia en la implementación de una red neuronal profunda mejorada a través de la plataforma de Nvidia DIGITS.

- De igual manera, se logró entrenar la red neuronal con las herramientas necesarias acorde a las especificaciones que se observa en la sección 3.1 de Requerimientos. Esta tecnología agilizó el proceso de entrenamiento de nuestra red neuronal optimizada con imágenes previamente seleccionadas y con requisitos específicos como calidad y tamaño de imagen, obteniendo resultados óptimos en validación de modelo.

- Se desarrollo un aplicativo web tanto para la gestión y administración de usuarios precisando en los módulos de roles de usuario, registro de denunciante y desaparecido, siendoeste último de gran importancia porque tendrá toda la información de la persona, principalmente las imágenes a clasificar y el código de búsqueda, cuyos datos son necesarios para la generación de la red neuronal. Adicional a ello se agregó un módulo de alertas que notificará al usuario la información de que una persona fue posiblemente localizada.

- Se logró desplegar en un servidor un aplicativo web, que permite la gestión y administración de registros de personas desaparecidas y la interacción con la cámara de la JetsonTX2 y el servidor con Nvidia DIGITS todo desde la misma interfaz. Adicional se utilizó como complemento la api de DIGITS y los servicios de SFTP y SSH para lograr el intercambio de datos de una manera segura y eficiente.

- Tal como se aprecia en la figura 4.7 y 4.8, no existe inconveniente en la inserción y listado

de los registros. Además de ello, en las figuras 4.13 y 4.14 se puede corroborar que la transferencias de archivos entre el aplicativo web y la jetson TX2 fueron exitosas. Finalmente en la figura 4.14, se demuestra que el consumo de recursos durante los procesos mencionados son bajos, dado que el consumo de memoria es de apenas 1/3 de su capacidad total, al igual que el consumo del procesador que apenas llega a un 2 % de su capacidad.

RECOMENDACIONES

- Al implementar una red neuronal se debe observar las especificaciones del conjunto de datos del que se va a trabajar ya que influye diversos factores como las (épocas, tamaño del lote) para el rendimiento de la misma.
- Se recomienda de preferencia desplegar los modelos entrenados en la carpeta usuario para evitar problemas de permiso en otras rutas de directorio.
- Se recomienda utilizar las librerías de visión artificial estables acordes a la versión del sistema operativo; dado que las últimas versiones de librerías son incompatibles con versiones de sistemas operativos Linux antiguos; tanto en el módulo Jetson TX2 como el servidor con Nvidia DIGITS
- Para una mejor predicción se debe considerar que a mayor cantidad de categorías por clasificar, mayor debe ser el tamaño de lote; y a mayor cantidad de imágenes por categoría,el porcentaje de predicción será más exacto.
- El Api Rest de Nvidia DIGITS trabaja conjuntamente con el comando cURL. Se recomienda verificar si está instalado esta librería dado que no viene instalado en todos los sistemas basados en UNIX.
- Para mejorar la seguridad, se recomienda cambiar la configuración de puertos que vienen instalados por defecto en los servicios SSH y SFTP; además de limitar el número de intentos de logueo o solamente permitir la conexión remota con las credenciales para un usuario especifico.

REFERENCIAS

[1] R. Solé, "Qué son los nvidia cuda cores," 2019. [Web; accedido el 11-07-2020].

[2] A. Geitgey, "Build a hardware based face recognition system with the nvidia jetson nano and python," 2019.

[3] E. Haines, "Real-time object detection in 10 lines of python code on jetson nano [archivo de video]," 2020.

[4] R. M. Castillo Pérez, "Responsabilidad estatal en la desaparición forzada de personas," B.S. thesis, 2016.

[5] H. Gutama, "Unos 200 menores, el 70 % niñas, desaparecieron durante la pandemia en ecuador," *Diario El Mercurio*, 2020. [Web; accedido el 11-07-2020].

[6] D. N. M. García, V. M. D. Flores, J. L. H. López, E. I. A. Jiménez, and E. F. V. Acurio, "Avances de la inteligencia artificial en salud," *Dominio de las Ciencias*, vol. 5, no. 3, pp. 603–613, 2019.

[7] APD, "¿qué es machine learning y cómo funciona?," 2019. [Web; accedido el 11-07-2020].

[8] A. Cifuentes, E. Mendoza, M. Lizcano, A. Santrich, and S. Moreno-Trillos, "Desarrollo de una red neuronal convolucional para reconocer patrones en imágenes," *Investigación y desarrollo en TIC*, vol. 10, no. 2, pp. 7–17, 2019.

[9] C. X. Niola Quito and W. A. Sanango Zhinin, "Desarrollo de un software de seguridad para detección y reconocimiento facial basado en los algoritmos de viola-jones y pca eigenface," B.S. thesis, 2019.

[10] D. E. E. Olguín and P. I. J. Guillen, "Reconocimiento facial," *Pontificia Universidad Católica De Valparaíso Facultad De Ingeniería Escuela De Ingeniería Informática*, 2015.

[11] A. Gebhart, "Reconocimiento facial: Apple, amazon, google y la carrera por captar tu cara," 2019.

[12] Kimaldi, "Reconocimineto facial," 2014. [Web; accedido el 11-07-2020].

[13] R. Uribe-Paredes, D. Cazorla, E. Arias, and J. L. Sánchez, "Un sistema heterogéneo multicore/gpu para acelerar la búsqueda por similitud en estructuras métricas," *Ingeniare. Revista chilena de ingeniería*, vol. 22, no. 1, pp. 26–40, 2014.

[14] M. F. Piccoli, “Computación de alto desempeño en gpu,” 2011.

[15] J. Gomar, “Nvidia cuda cores y cual es su importancia,” 2018. [Web; accedido el 11-07-2020].

[16] L. Raffin, “70 aplicaciones que utilizan aceleracion por gpu,” 2012. [Web; accedido el 11-07-2020].

[17] M. Janakiram, “Nvidia digits an easy way to get started with deep learning,” 2018.

[18] S. Arteaga, “Nvidia jetson tx2, el superordenador de ia del tamaño de una tarjeta,” 2017. [Web; accedido el 11-07-2020].

[19] J. P. Balarini, “Facial recognition using neural networks over gpgpu,” *CLEI Electronic Journal*, vol. 15, no. 3, pp. 6–6, 2012.

[20] J. A. Valderrama Molano *et al.*, “Clasificación de objetos usando aprendizaje profundo implementado en un sistema embebido,” Master’s thesis, Universidad Autónoma de Occidente, 2017.

[21] F. López-Saca *et al.*, “Clasificación de imágenes usando redes neuronales convolucionales,” Master’s thesis, Universidad Autónoma Metropolitana (México). Unidad Azcapotzalco . . . , 2019.

[22] Nvidia-Corporation, “Gpu accelerated computing with c and c++,” 2020.

[23] A. Martins, “Qué es el .ªprendizaje profundo"de la inteligencia artificial y cómo ya está cambiando la vida de millones de personas en todo el mundo,” 2017. [Web; accedido el 11-11-2021].

[24] NVIDIA, “Deep learning digits documentation,” 2021.

[25] S. G. Chagcha Freire, “El sistema operativo linux ubuntu y su incidencia en la enseñanza aprendizaje en el centro educativo bautista en el año 2010,” B.S. thesis, 2012.

[26] M. A. E. Mina and A. Y. S. Cedeño, “Análisis comparativo entre asp. nety php,” *INNOVA Research Journal*, vol. 3, no. 4, pp. 25–43, 2018.

[27] R. J. Acosta Morales *et al.*, *Reconocimiento de placas vehiculares aplicando procesamiento de imágenes digitales en Python-OpenCV*. PhD thesis, 2020.

[28] J. C. Cuervo Restrepo and H. J. Chavarro Hurtado, “Aplicación de algoritmos de deep learning en un sistema embebido para el control de una mano robótica,” 2018.

ANEXOS

ANEXO A

APLICACIÓN WEB

En la siguiente figura 0.1 podemos observar la interfaz para el inicio de sesión en el cual debemos ingresar nuestro usuario y contraseña.

Figura 0.16: Iniciar Sesión

En la figura 0.2 podemos observar el menú del aplicativo web con diferentes modulos

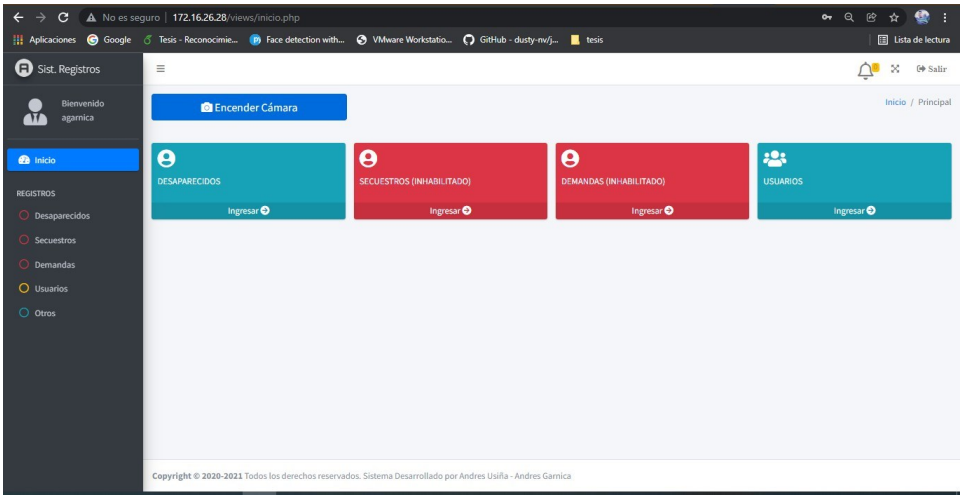


Figura 0.17: Modulo Usuario

En la figura 0.3 tenemos el módulo de ingresar los datos de los usuarios como por ejemplo nombres, cédula, nacionalidad entre otros.

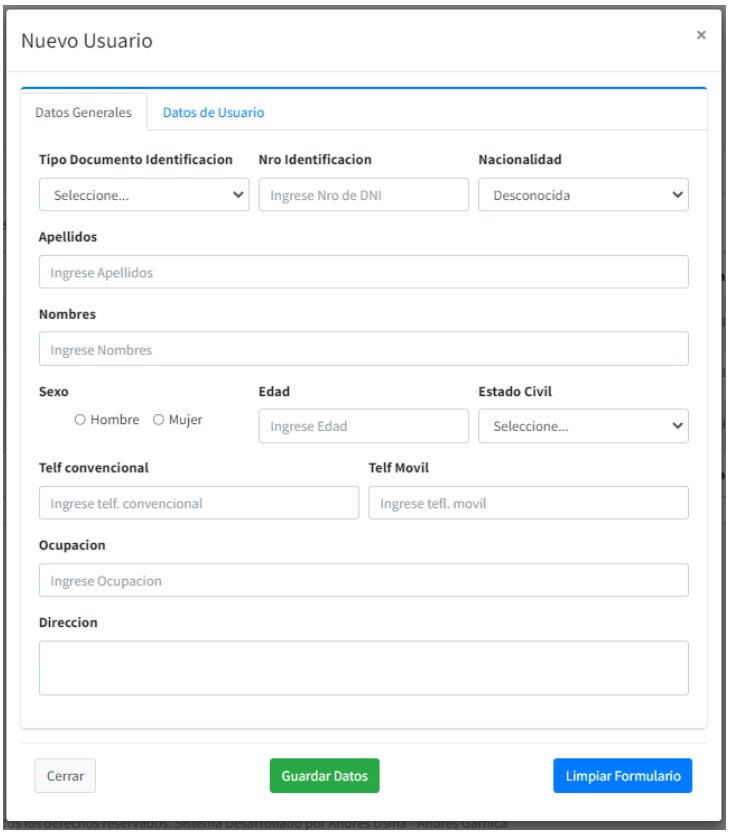


Figura 0.18: Modulo usuario ingreso de datos generales

En la figura 0.4 tenemos el módulo donde ingresamos el perfil de usuario tanto en administrador o usuario.

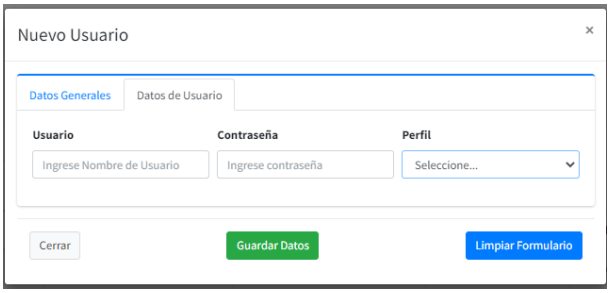


Figura 0.19: Creación del perfil del Usuario.

En la figura 0.5 aquí tenemos el módulo en donde se ingresan todos los datos del denunciante el cual se guardará en la base de datos, en donde el campo más importante es la fecha de la denuncia.

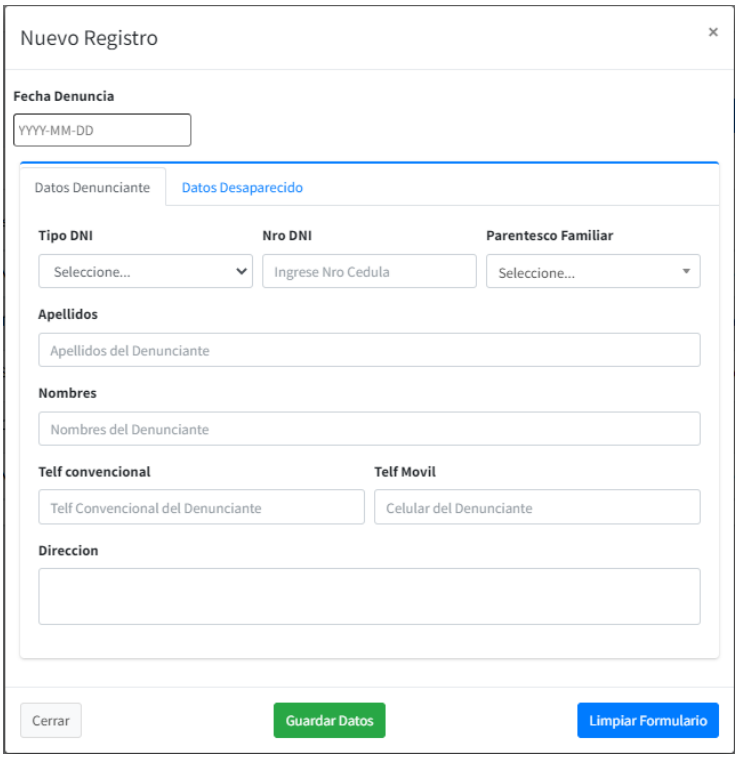


Figura 0.20: Creación del Módulo Denunciante.

En la figura 0.6 aquí tenemos el módulo en donde se ingresamos los datos del desparecido los campos de vital importancia son las imágenes y el código búsqueda ya que este es el que influye para el entrenamiento de la red en la plataforma DIGITS.

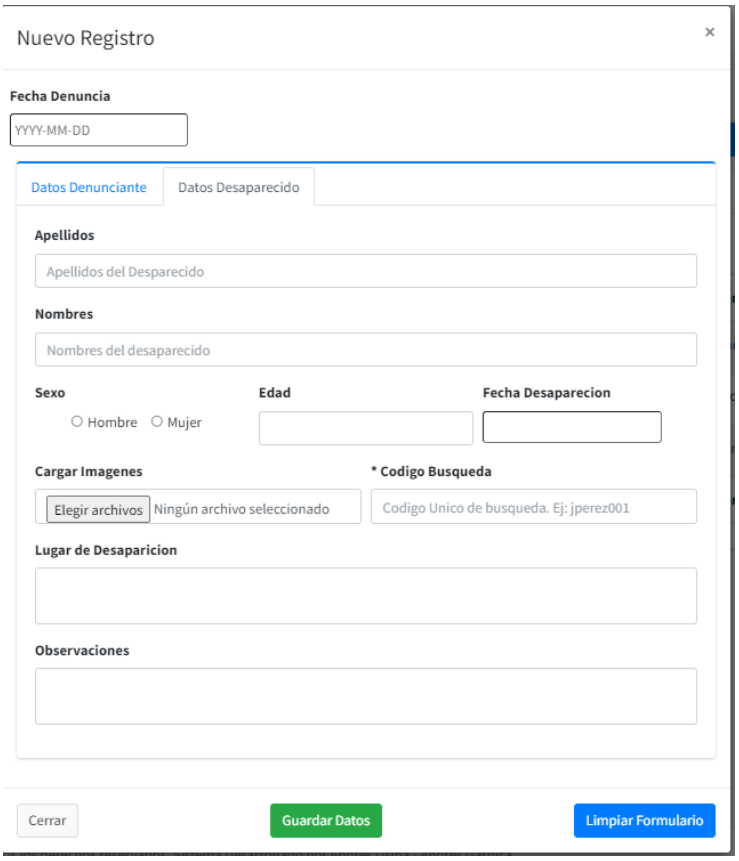


Figura 0.21: Creación del Módulo Desaparecido.

En la figura 0.7 encontramos la lista de Usuarios que contiene el aplicativo web como por ejemplo el administrador.

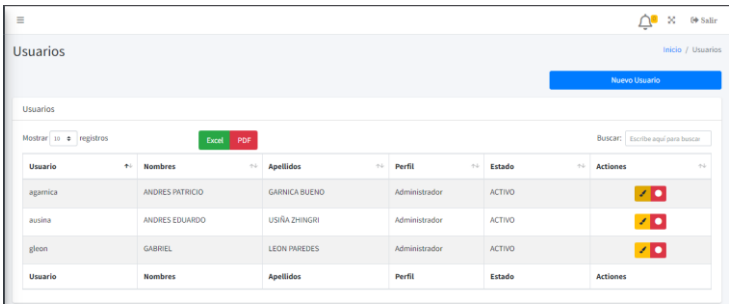


Figura 0.22: Listado de Usuarios.

Aquí observamos la figura 0.8 donde encontramos el listado de Desaparecidos dado que este es de gran importancia porque contienen toda la información para poder enfocarnos en el entrenamiento de nuestra red.

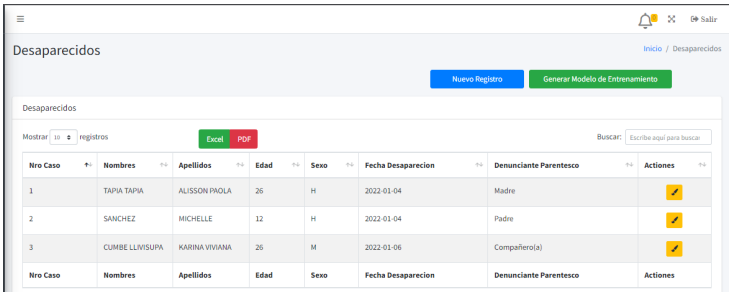


Figura 0.23: Listado de Usuarios.

A continuación, en la figura 0.9 tenemos un cuadro informativo en el cual realiza él envió de lasimágenes desde el aplicativo web hacia la plataforma de digits.

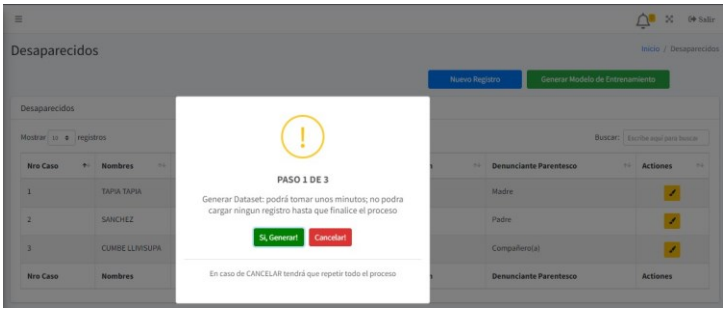


Figura 0.24: Envió de Imágenes a la Plataforma DIGITS.

En tanto en esta figura 0.10 observamos los resultados, es decir todas las imágenes se enviaron a la Plataforma DIGITS a su carpeta raíz, lo que sucedió internamente se realizó en la figura 0.9

Figura 0.25: Proceso Interno del Aplicativo WEB y DIGITS.

Siguiendo la secuencia del punto anterior ahora observaremos en la figura 0.11 la creación del dataset este se encarga de procesar todas las imágenes a través de la API, en la figura 0.11 ya se muestra que el dataset se creó correctamente una vez terminado empieza la creación del modelo, así como se observa en la figura 0.12 en el cual este se demorara dependiendo el tamaño de los datos.

Figura 0.26: Generar Dataset a través del Api Rest.

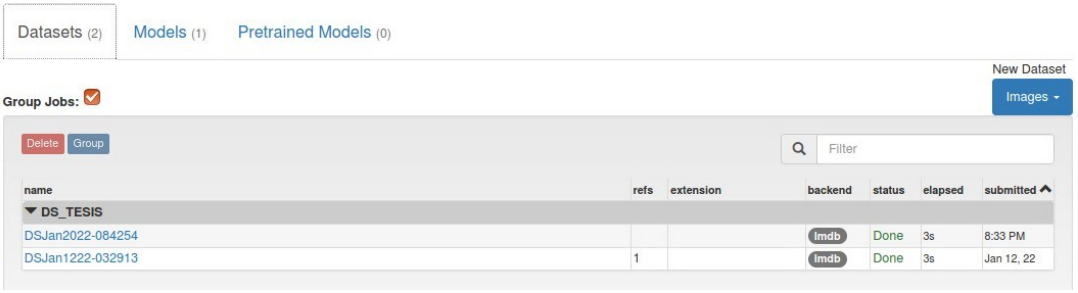


Figura 0.27: Dataset creado.

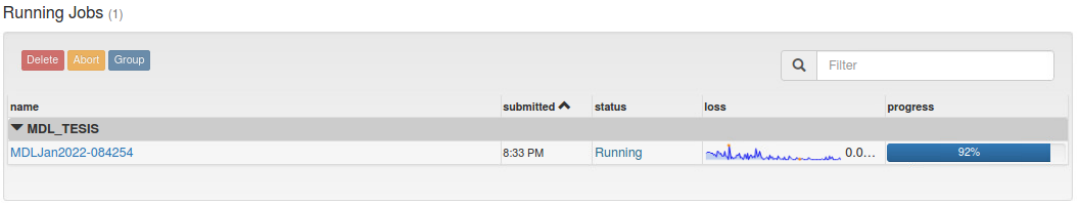


Figura 0.28: Ejecución de modelo.

Figura 0.29: Modelo Creado .

Por último, tenemos la descarga de nuestro modelo como observamos en la figura 0.15 y 0.16, el cual se realizará a través de nuestro aplicativo este será hará a través del servicio de FTP en donde los archivos se envían desde el DIGITS hacia el aplicativo.

Figura 0.30: Proceso de Descarga de Archivos

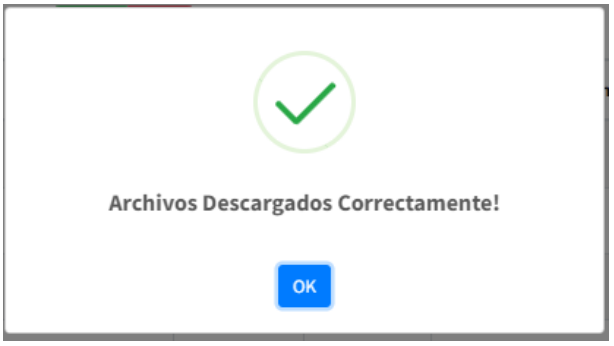


Figura 0.31: Mensaje de los Archivos Descargados.

Los resultados se pueden evidenciar en la figura 0.17 donde aquí observamos que todos los archivos de nuestra red neuronal entrenada se encuentran en el servidor web y este enviara a la Jetson TX2 a través del servicio de FTP.

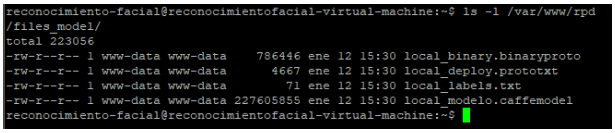


Figura 0.32: Archivos Descargados.

ANEXO B

ENTRENAMIENTO DE LA RED NEURONAL PROFUNDA EN LA PLATAFORMA DIGITS

En primera instancia para crear nuestra red neuronal profunda a través de la plataforma de Nvidia Digits, debemos iniciar sesión como observamos en la figura 0.18 posterior a ello nos aparecerá un menú como se indica en la figura 0.19.

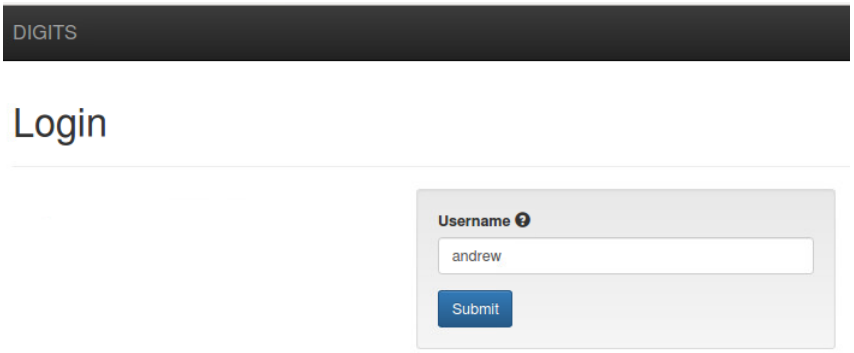


Figura 0.33: Inicio de Sesión.

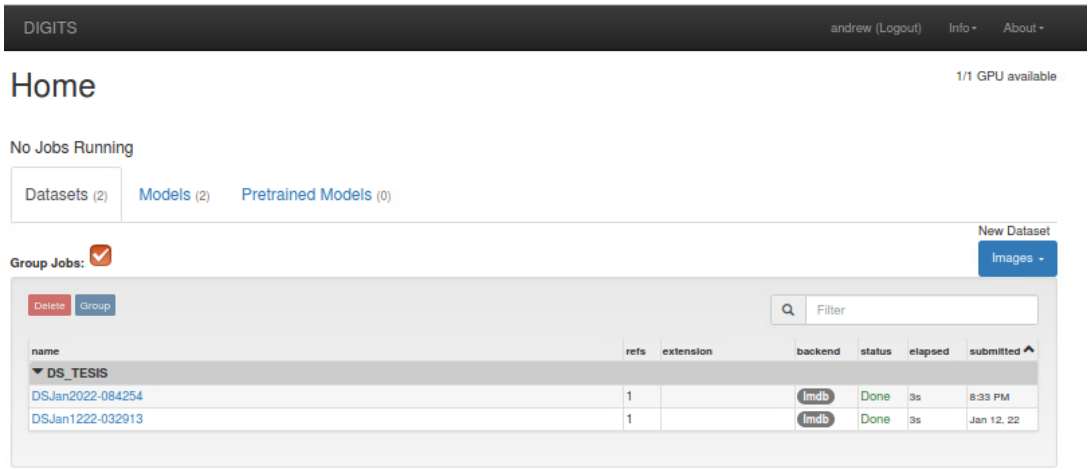


Figura 0.34: Menú de DIGITS.

De manera, que en este submenú que se observa en la figura 0.20, aquí tendremos que escoger nuestra opción para el entrenamiento en nuestro caso selecciona la clasificación ya que nosotros en este proyecto estamos enfocados en la clasificación de rostros a través de imágenes.

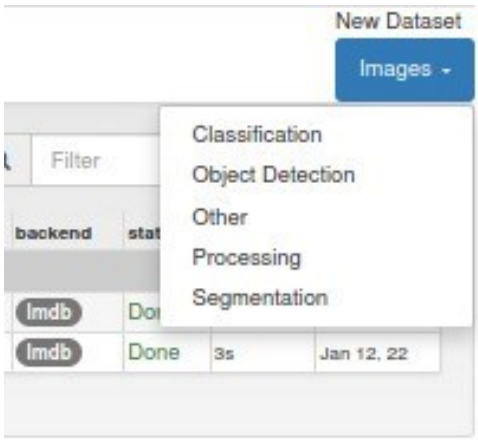


Figura 0.35: Submenú.

Una vez seleccionado nos aparecerá lo siguiente una ventana en donde indica que se va a realizar la creación del dataset, así como se observa en la figura 0.21, en donde el campo de gran importancia es la ruta en donde están almacenada las imágenes y el tipo de imagen otros campos relevantes son el nombre del dataset

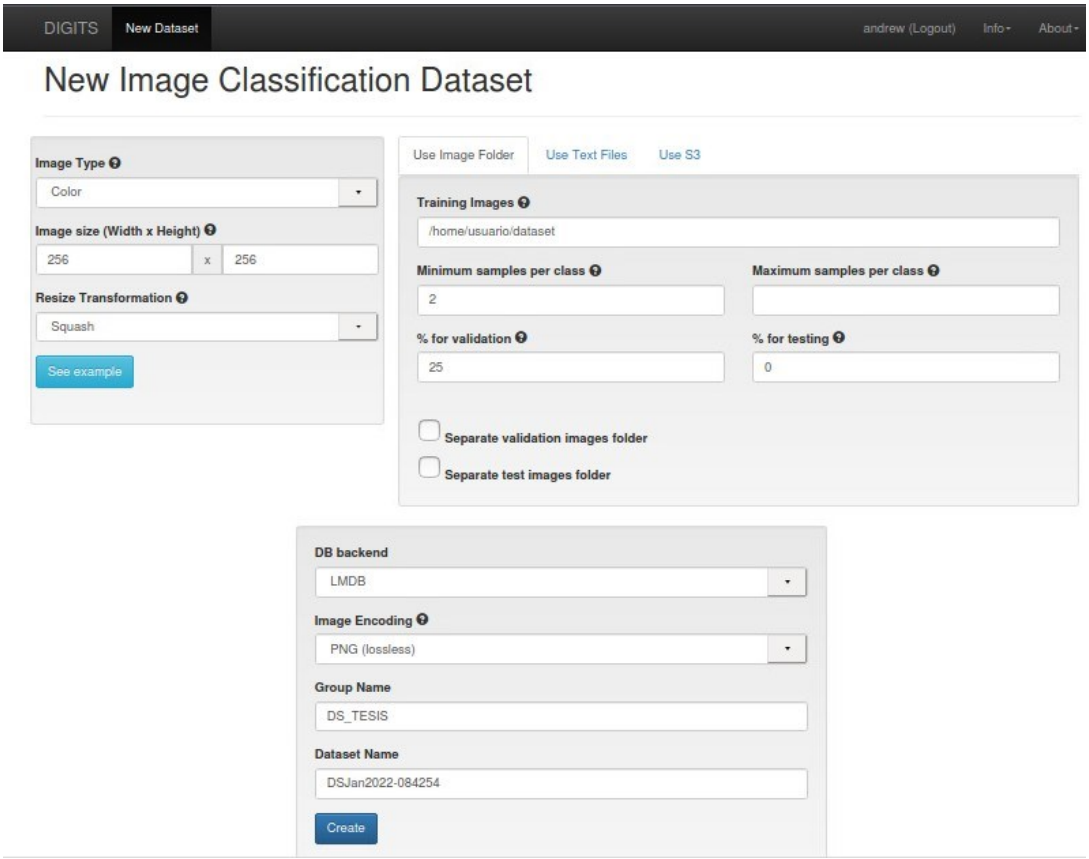


Figura 0.36: Crear Dataset.

Para la creación del modelo primero debemos crear el conjunto de datos, para así poder seleccionar el mismo y dependiendo del tamaño vamos colocar el número de épocas y tamaño de lote ya que estos influyen mucho para la creación de la red neuronal profunda.

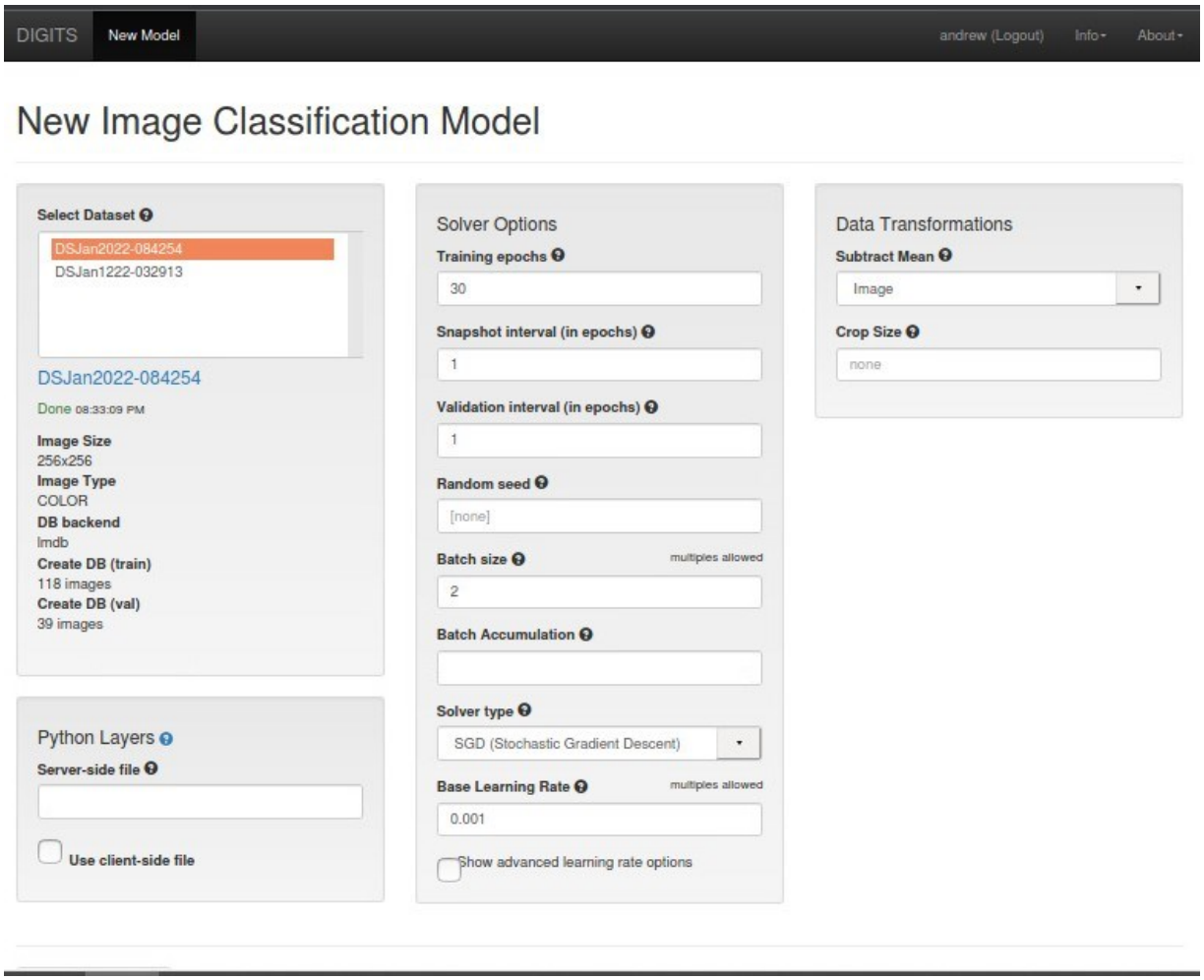


Figura 0.37: Campos a llenar para creación del Modelo.

En la segunda parte de la creación del modelo es muy importante seleccionar una red estándar en nuestro caso seleccionaremos ALEXNET dado que esta trabaja en el tipo de imagen y tamaño con el que se creó nuestro conjunto de datos como se observa en la figura 0.22.

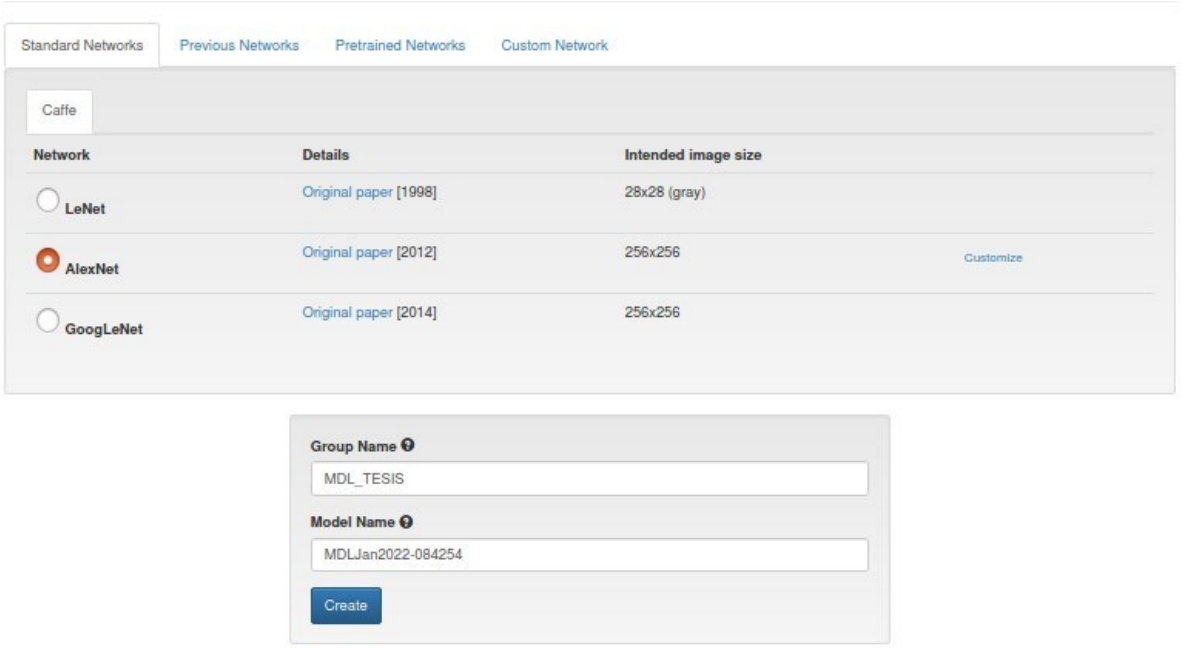


Figura 0.38: Selección de la Red Estandarizada para crear el modelo.

En la siguiente figura se muestra como nuestra red va acercándose a la predicción que deseamos y también la pérdida de tiempo real. Con esto podremos observar que tan eficiente es nuestra red.

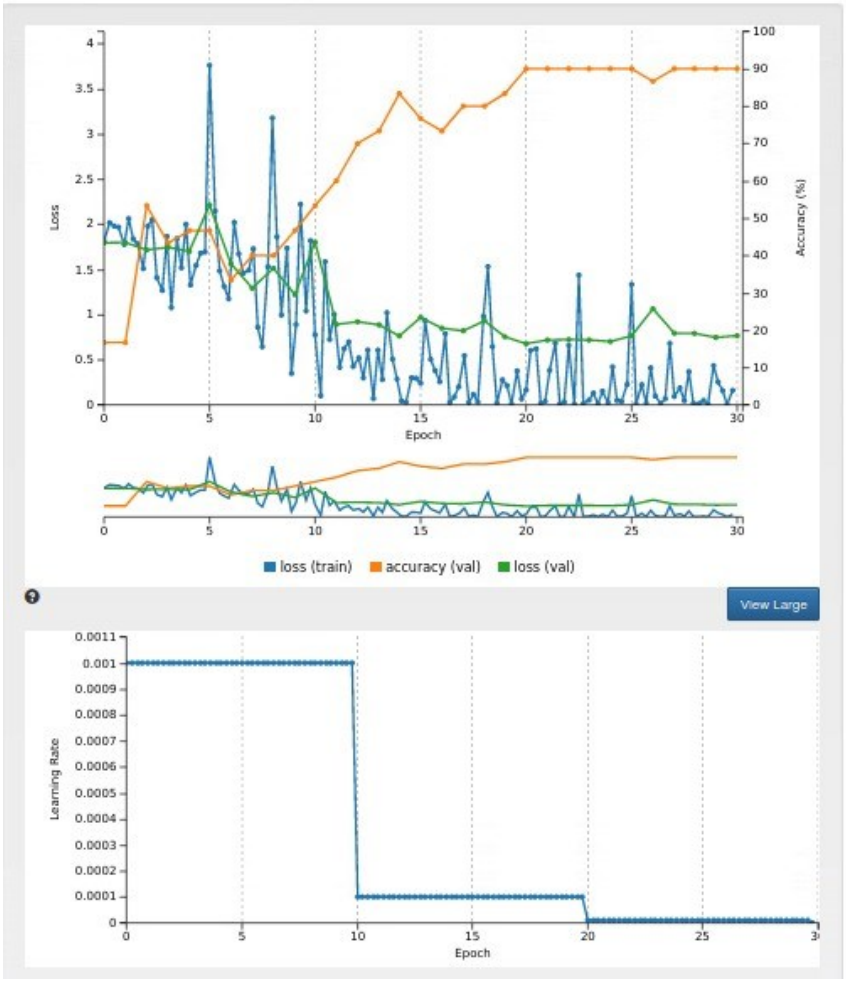


Figura 0.39: Perdidas y Predicción

Por último, se realizará pruebas dentro de la plataforma DIGITS el cual hemos escogido de la persona ANDRÉS EDUARDO USIÑA ZHINGRI y los resultados fueron óptimos ya que se alcanzó una predicción del 99.99 % como se observa en la figura 0.25. Pero hay que tener en cuenta que para una red neuronal sea óptima se debe tener calidad de imagen como un punto muy relevante.

Figura 0.40: Resultados del Modelo creado.

ANEXO C

DESPLIEGUE DEL MODELO EN EL SUPERORDENADOR Nvidia Jetson TX2 Y NOTIFICACIONES

Para el despliegue de nuestro modelo en el superordenador se va ir indicando fragmentos del código que se utilizó para la ejecución de la misma.

Este es el mismo proceso el que se realizó para lo de la plataforma DIGITS, pero en cambio ahora solo cambiamos la dirección IP a la que desea enviar en este caso será a la del superordenador Nvidia Jetson TX2.

```
if($generaDeploy && $generaCaffe && $generaBinary && $generaLabels){
    $checkArchivosDescargados = "ok";
    //echo "ok";
    if($checkArchivosDescargados == "ok"){
        //$ftp_svr ="192.168.20.25";
        $ftp_svr ="172.16.26.34";
        $ftp_usuario = "nvidia";
        $ftp_pass ="nvidia";
        $ftp_conn = ftp_connect($ftp_svr);
        $login = ftp_login($ftp_conn,$ftp_usuario, $ftp_pass);

        if(!$ftp_conn || (!$login)){
            echo "no se pudo conectar";
            exit;
        }else{

            $carpetaLocal = "../files_model"; //carpeta local que tien
//$carpetaRemota = 'pruebas08'; //carpeta donde se guardara los archivos model

$a = array_map("htmlspecialchars", scandir($carpetaLocal));
$c = 0;
```

Figura 0.41: Conexión FTP para descarga de archivos.

Para la ejecución de la cámara a través del aplicativo se la realizo mediante un servicio de SSH, se logró la conexión al dispositivo y también a su vez se crea un archivo remoto el cual tendrá todos los permisos y el cual se ejecutará para poder encender la cámara y eso se observará a continuación en la figura 0.27

```php
echo "GENERACION DE BASH CON CONTENIDO ". "<br><br>";
$connection = ssh2_connect('172.16.26.34',22);
$conexion =ssh2_auth_password($connection,'nvidia','nvidia');
//$checkConexion=ssh2_auth_password($conexion);
if($conexion){
    echo "CONEXION CORRECTA\n\n"."<br>";
    $linea1="#!bin/bash \n";
    $linea2="export DISPLAY=:0; python3 tegra-cam-caffe.py --crop
    --usb --vid 1 --prototxt local_deploy.prototxt --model local_modelo.caffemodel --labels local_labels.txt
    --mean local_binary.binaryproto --output softmax";
    $stream= ssh2_exec($connection, "echo $'".$linea1.$linea2."'>test5.sh; chmod +x test5.sh; bash test5.sh");
   // $stream = ssh2_exec($connection, './pruebaJetson.sh');
stream_set_blocking($stream, true);

echo "Resultado de la ejecucion del archivo es:  "."<br>";
echo "==================================";
$stream_out =ssh2_fetch_stream($stream,SSH2_STREAM_STDIO);
echo "<pre>";
echo stream_get_contents($stream_out);
echo "<pre>";
}else{
    echo "FALLO CONEXION";
}
```

Figura 0.42: Código Ejecución.

Desde la parte gráfica tenemos un botón para el despliegue en donde se encenderá la cámara y comenzara la detección de rostros y al tener un porcentaje mayor al 90 % empezara a llegar las notificaciones de que dicha persona fue localizada.

Figura 0.43: Encender Cámara.

Mediante el paquete requests de Python se realizó las notificaciones el cual esta se encarga de enviar los parámetros a continuación se observa la línea de código utilizada.

```
if var >=90 and top_labels[i]!='datos':
   print("Persona OK")
   #res=requests.post("http://192.168.1.20/ajax/conexion_fast.php", data=keys)
   res=requests.post("http://172.16.26.28/ajax/jetson_notificacion.ajax.php", data=keys)
   #res=requests.post("http://192.168.1.35/test/hola.php", data=keys)
   pastebin_url=res.text
   print(res.url)
   ##print(res.data)
   print(res.status_code)
   text=res.text
   print(text)
   print(pastebin_url)
elif var<=89:
   print("Persona no OK")
```

Figura 0.44: Conexión Python con PHP.