Identificación de Incendios Forestales en Imágenes Satelitales usando Machine Learning/Deep Learning apoyado en Sistemas Computacionales Integrados

Jhon Deivy Pérez Arguello

Director

Julián Gustavo Rodríguez Ferreira

Doctor en Física, Especialidad Astrofísica

Codirector

Carlos Jaime Barrios Hernández

Doctor en Informática

Universidad Industrial de Santander

Maestría en Ingeniería de Sistemas e Informática

Facultad de Ingenierías Físico-Mecánicas

Escuela de Ingeniería de Sistemas e Informática

Bucaramanga

2023

Dedicatoria y Agradecimientos

A mi familia por acompañarme y apoyarme en este reto personal.

A todos los docentes y en especial a mis directores de proyecto por su conocimiento, dedicación y paciencia.

A todas las personas que de alguna manera hicieron posible que este proyecto se efectuara exitosamente.

Contenido

Pág.

Lista de Figuras

Pág.

Lista de Tablas

Pág.

Resumen

Título: Identificación de Incendios Forestales en Imágenes Satelitales usando Machine Learning/Deep Learning apoyado en Sistemas Computacionales Integrados[*]

Autor: Jhon Deivy Pérez Arguello[**]

Palabras Clave: Aprendizaje automático, Redes Neuronales Convolucionales, Sistemas Embebidos, Visión por computador.

Descripción:
La detección temprana de incendios forestales es un aspecto crítico en la lucha contra los desastres naturales que involucra a empresas, instituciones académicas y gobiernos. La inteligencia artificial puede ser una herramienta valiosa en la tarea de identificación de incendios forestales y en la promoción del uso de tecnologías avanzadas en los procedimientos actuales, pero su aplicación requiere una gran capacidad computacional y energética. Con este objetivo, se presenta un modelo de identificación de incendios forestales en imágenes satelitales de alta resolución que es altamente confiable y eficiente, y puede ser ejecutado en un sistema embebido. Se creó un Dataset de imágenes satelitales con bandas espectrales "M3I3M11", se entrenaron dos redes neuronales con este Dataset y se desarrolló un sistema de visión artificial llamado IGNIS, que funciona sin necesidad de entrenamiento previo y utiliza la biblioteca OpenCV. Para elegir el mejor de los tres algoritmos, se ejecutaron en el hardware embebido Jetson Nano de Nvidia, evaluando factores como la temperatura del dispositivo, el consumo energético, la memoria RAM, la GPU y la confiabilidad de las inferencias. La evaluación demuestra que el algoritmo IGNIS obtuvo el mejor puntaje en términos de confiabilidad con un 98% y en términos de consumo de recursos con un 29%.

[*] Trabajo de Grado

[**] Facultad de Ingenierías Físico-Mecánicas. Escuela de Ingeniería de Sistemas e Informática. Maestría en Ingeniería de Sistemas e Informática. Director: Julián Gustavo Rodríguez Ferreira, Doctor en Física, Especialidad Astrofísica. Codirector: Carlos Jaime Barrios Hernández, Doctor en Informática.

Abstract

Title: Identification of Forest Fires in Satellite Images Using Machine Learning / Deep Learning Supported in Integrated Computer Systems*

Author: Jhon Deivy Perez Arguello**

Keywords: Machine Learning, Convolutional Neural Networks, Embedded Systems, Computer Vision.

Description:
The early detection of forest fires is a critical aspect in the fight against natural disasters that involves companies, academic institutions and governments. Artificial intelligence can be a valuable tool in the task of identifying forest fires and in promoting the use of advanced technologies in current procedures, but its application requires great computational capacity and energy. With this objective, a model for the identification of forest fires in high-resolution satellite images is presented, which is highly reliable and efficient, and can be executed in an embedded system. A dataset of satellite images with spectral bands "M3I3M11" was created, two neural networks were trained with this dataset and an artificial vision system called IGNIS was developed, which works without prior training and uses the OpenCV library. To choose the best of the three algorithms, they ran on Nvidia's Jetson Nano embedded hardware, evaluating factors such as device temperature, power consumption, RAM, GPU, and reliability of inferences. The evaluation shows that the IGNIS algorithm obtained the best score in terms of reliability with 98% and in terms of resource consumption with 29%.

* Degree work

** Faculty of Physical-Mechanical Engineering. School of Systems Engineering and Informatics. Master's Degree in Systems Engineering and Informatics. Director: Julián Gustavo Rodríguez Ferreira, PhD in Physics, Astrophysics Specialty. Co-director: Carlos Jaime Barrios Hernández, PhD in Computer Science

Introducción

Los incendios forestales son una de las principales causas de degradación ambiental y pérdida de biodiversidad a nivel global. Su impacto puede ser devastador tanto a nivel económico como social y ambiental, por lo que es importante tomar medidas para prevenir y detectar incendios de manera temprana.

Una forma de detectar incendios tempranamente es a través del uso de imágenes satelitales, que permiten observar grandes áreas de manera rápida y precisa (Petrescu & et.al., 2018). Sin embargo, el procesamiento y análisis de estas imágenes requiere un gran poder computacional y puede resultar costoso y en algunos casos poco eficientes.

Por esta razón, es necesario elaborar un modelo computacional que sea capaz de identificar incendios forestales en imágenes satelitales de manera eficiente y precisa. Además, este modelo debe ser capaz de funcionar en hardware embebido, es decir, dispositivos compactos y de bajo costo que puedan ser integrados en sistemas de monitoreo y detección de incendios.

Una de las tecnologías que puede utilizarse para crear este modelo es el deep learning (DL), una rama de la inteligencia artificial (IA) que permite a las computadoras "aprender" por sí mismas a partir de grandes conjuntos de datos. El DL es muy eficiente para el procesamiento y análisis de imágenes, por lo que es una opción prometedora para el modelo de detección de incendios.

Otra tecnología IA que puede utilizarse es la visión artificial, que se basa en el uso de algoritmos para procesar y analizar imágenes y video. La visión artificial es muy útil para identificar patrones y características en imágenes, y puede ser utilizada en conjunción con el DL para mejorar la precisión del modelo de detección de incendios.

En algunos casos, la IA se apoya en hardware embebido para llevar a cabo sus tareas de manera más eficiente (Brozek, 2014). Los dispositivos embebidos, también llamados placas de desarrollo integradas, son pequeños y compactos, y pueden ser integrados fácilmente en sistemas de monitoreo y control. Además, suelen tener un bajo consumo energético y un bajo costo monetario, lo que los convierte en una opción atractiva para la implementación de tecnologías de IA.

En general, desde el procesamiento de imágenes hasta el monitoreo ambiental, la IA puede proporcionar soluciones rápidas y precisas a problemas complejos (Feng, Jiang, Yang, Du, & Li, 2019).

En este trabajo, se presenta un modelo computacional que utiliza tecnologías de visión artificial y/o DL para identificar incendios forestales en imágenes satelitales y que puede ser implementado en hardware embebido. Se describirá el proceso de creación, selección y evaluación del modelo, así como su aplicación en la detección temprana de incendios forestales

1. Objetivos

1.1 Objetivo General

Elaborar un modelo con técnicas de *Machine Learning / Deep Learning* (ML/DL) en arquitecturas computacionales que garanticen el procesamiento de alto rendimiento en sistemas integrados que permita identificar incendios forestales a partir de imágenes satelitales.

1.2 Objetivos Específicos

Crear un Dataset de imágenes satelitales para ser utilizadas en el entrenamiento de la red neuronal y su respectiva inferencia.

Seleccionar y adaptar un algoritmo Deep learning que sea susceptible de implementarse en una placa de desarrollo integrada.

Establecer un workflow para el entrenamiento de la red neuronal, que pueda ser replicado por otro usuario.

Evaluar el desempeño del modelo identificando incendios en el dataset de imágenes satelitales.

2. Alcance

La herramienta que se desarrolla en este proyecto permite identificar incendios forestales en imágenes satelitales utilizando técnicas de aprendizaje automático y/o visión artificial. Se implementa en hardware embebido para, en posteriores proyectos, integrarla a dispositivos móviles de monitoreo climático, lo que permitirá realizar la identificación in situ, ahorrando en la transmisión de datos hacia las centrales de control y brindando una respuesta oportuna. Se proporciona un Dataset de imágenes satelitales para entrenar redes neuronales y un workflow para el entrenamiento e inferencias posteriores.

3. Marco Teórico

3.1 Inteligencia Artificial

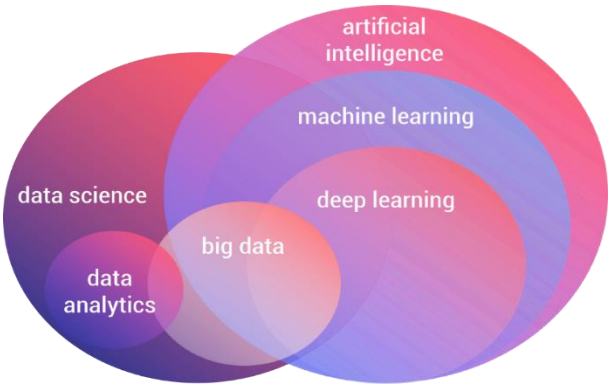
La Inteligencia Artificial (IA) es una disciplina académica relacionada con la teoría de la computación cuyo objetivo es emular algunas de las facultades intelectuales humanas en sistemas artificiales (Meseguer González, López, & Badia, 2017). Estas facultades abarcan la percepción sensorial del entorno y la interpretación de este. La IA se ha implementado en diferentes campos como juegos, demostraciones matemáticas, traducción de texto, etc., pero no todas las actividades en las que el ser humano necesite realizar un proceso lógico de pensamiento son susceptibles de emular.

Es indudable que la influencia de otras disciplinas como la estadística, el cálculo matemático, la psicología, la robótica, la neurociencia, entre otras, es fundamental para el desarrollo y evolución de la IA. La ilustración 1 muestra la relación entre la inteligencia artificial,

el aprendizaje automático y el aprendizaje profundo. La inteligencia artificial es el concepto más amplio y engloba a los otros dos, mientras que el aprendizaje automático es una rama de la inteligencia artificial que incluye tanto el aprendizaje como el aprendizaje profundo. Por último, el aprendizaje profundo es una técnica de aprendizaje automático que utiliza redes neuronales y otras técnicas para permitir que las computadoras aprendan de manera más profunda y detallada. Además, la IA también aporta transversalmente a otras áreas científicas como la ciencia de datos y lo que ella contiene (Benítez, 2014).

Figura 1.

Campos Relacionados y Aplicaciones de la Inteligencia Artificial.



Nota: miro.medium. (sf). **Campos Relacionados y Aplicaciones de la Inteligencia Artificial** https://miro.medium.com/max/1200/1*24Hj2PgyawR20kJBj8NU0g.png

3.2 Machine Learning

Machine Learning (ML) es un conjunto de técnicas y algoritmos que permiten que una máquina aprenda de forma automática a partir de experiencias pasadas (Casas Roma, Bosch Rué, & Lozano Bagén, 2019). En este proceso, se crea un modelo de un problema en particular que puede ser solucionado por medio de un algoritmo inteligente que se alimenta con datos para su entrenamiento. Los resultados del modelo son comparados con la respuesta ideal y, a través de iteraciones, se ajustan para mejorar la precisión. El entrenamiento puede ser supervisado o no

supervisado.

El entrenamiento supervisado es una técnica de aprendizaje automático en la que se utilizan conjuntos de datos etiquetados, es decir, que vienen acompañados de la respuesta deseada. El objetivo es que la máquina aprenda a mapear entradas a salidas, en otras palabras, a tomar decisiones correctas sobre el conjunto de datos de entrenamiento y aplicarlas al conjunto de datos de prueba. Por otro lado, el entrenamiento no supervisado es una técnica de aprendizaje automático en la que se utilizan conjuntos de datos sin etiquetar, es decir, sin la respuesta deseada. En este caso, el objetivo es que la máquina descubra patrones o relaciones ocultas en los datos, sin una respuesta objetiva a la que comparar. Se busca que la máquina encuentre estructuras ocultas en los datos y las utilice para tomar decisiones. Ambos tipos de entrenamiento tienen aplicaciones prácticas en diversos campos, desde el procesamiento del lenguaje natural hasta el análisis de imágenes y el aprendizaje por refuerzo.

3.3 Deep Learning

Deep Learning, aprendizaje profundo o DL es el subcampo del ML que se dedica a construir algoritmos que explican y aprenden un amplio nivel de abstracciones de datos que los algoritmos tradicionales de aprendizaje automático a menudo no pueden. Los modelos de aprendizaje profundo normalmente se inspiran en fuentes de conocimiento, como la teoría de juegos o neurociencia y muchos de los modelos imitan la estructura básica de un sistema nervioso humano (Beysolow II, 2017).

El DL según su arquitectura, técnica y enfoque puede clasificarse en varios modelos de aprendizaje organizados en tres grupos: generativo, discriminativo e híbrido. El aprendizaje generativo se centra en la construcción de un modelo que se ajuste a los datos de entrada y pueda predecir el resultado esperado. Este tipo de modelo se basa en la probabilidad y en la construcción

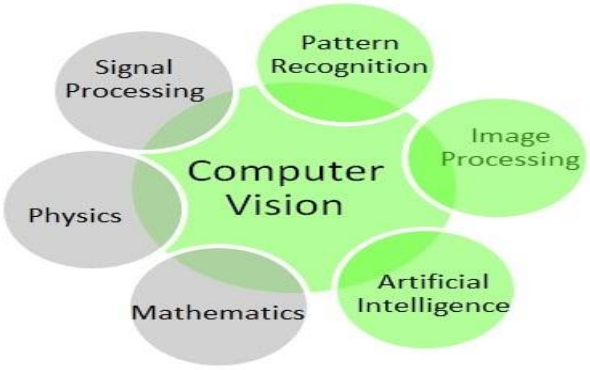
de una distribución que represente a los datos. El aprendizaje discriminativo, por otro lado, se centra en la separación de clases de datos a través de una función de decisión. Este tipo de modelo se basa en la minimización de la pérdida, es decir, en la diferencia entre la predicción y el resultado esperado. Finalmente, el aprendizaje híbrido combina ambas técnicas y es utilizado en casos donde es necesario tener en cuenta tanto la distribución de los datos como la separación de clases.

3.4 Visión Artificial

La visión artificial es un campo de la inteligencia artificial que se centra en desarrollar técnicas y algoritmos que permiten a las máquinas procesar e interpretar imágenes de manera similar a cómo lo hace el ser humano (Huang T. , sf, págs. 21–25.). Esto se logra a través de la implementación de sistemas de procesamiento de imágenes y el análisis de patrones, apoyado en los aportes de otras ciencias como la matemática y la física (ver ilustración 2).

Figura 2.

Campos Relacionados a la Visión por Computadora.



Nota: miro.medium. (sf). Campos Relacionados a la Visión por Computadora https://miro.medium.com/max/462/1*RmH092DDfJmAHwT7tdX3nA.jpeg

La visión artificial es esencial para automatizar tareas que implican la interpretación de imágenes y puede ser aplicada en una amplia variedad de campos como los siguientes:

- Detección de objetos (Maity A. , 2015): cuando el ordenador recibe imágenes en una

banda del espectro, puede relacionarla con patrones aprendidos con anterioridad e identificar los objetos.

- Análisis de video (Bruijning, Visser, Hallmann, Jongejans, & Golding, 2018): gracias a la detección de objetos se puede realizar un análisis a lo que ve el ordenador y automatizar las acciones en concordancia a los sucesos observados.

- Visión 3D (Soltani A. , Huang, Wu, Kulkarni, & Tenenbaum, 2017): proporciona al ordenador de la capacidad de emular la visión humana, siendo capaz de generar modelos tridimensionales de una imagen bidimensional usando diferentes técnicas.

3.5 Redes Neuronales Artificiales

Las Redes Neuronales Artificiales son sistemas de procesamiento de información que intenta emular el comportamiento con las redes neuronales biológicas (Caicedo Bravo & López Sotelo, 2009). Se han desarrollado generalizando modelos matemáticos teniendo en cuenta que:

- El procesamiento de información se realiza en elementos simples llamados neuronas.
- Las señales son pasadas entre neuronas a través de conexiones.
- Cada conexión tiene un peso asociado.
- Cada neurona tiene una función de activación que aplica a las señales de entrada para posteriormente generar una señal de salida

Las redes neuronales artificiales presentan características muy particulares que permiten su aplicación en diversos campos de desarrollo. Algunas de estas características son: la capacidad de aprendizaje apoyada en ejemplos que representen el problema a solucionar; la capacidad de generalización cuando se necesita dar una solución adecuada a un problema, utilizando datos de entrada que no se han utilizado en el proceso de aprendizaje; la capacidad para extraer características esenciales de los datos y rechazar lo que no sea relevante; capacidad de asociación

al establecer relación entre dos conceptos separados, entre otras.

3.6 Redes Neuronales Convolucionales

Una red neuronal convolucional o CNN es una red neuronal artificial profunda y alimentada en la cual la red neuronal preserva la estructura jerárquica al aprender representaciones de características internas y generalizar las características en los problemas comunes de imagen como el reconocimiento de objetos y otros problemas de visión por computadora. No está restringido a imágenes; También logra resultados de vanguardia en problemas de procesamiento del lenguaje natural y reconocimiento de voz. Una CNN consta de múltiples capas (Manaswi, 2018).

Con la teoría presentada se da paso a la segunda sección en la que se expone parte del estado del arte.

4. Estado del Arte

En esta sección se presenta parte de la evolución de la teoría planteada en el marco teórico y como puede ser usado en el desarrollo de la investigación. El análisis del estado del arte se enfocó en tres áreas de indagación, ya que el objetivo principal es resultado de la convergencia entre estas áreas:

1. La identificación de fuego en imágenes satelitales.
2. La identificación de elementos con técnicas de DL.
3. El despliegue de técnicas de DL sobre arquitecturas computacionales embebidas.

Es importante mencionar que, en el proceso de investigación para este trabajo, se consideraron varias opciones para elaborar el estado del arte, pero debido a la gran cantidad de

información disponible, es posible que no estén incluidos otros proyectos que también son relevantes en este contexto.

En el estado del arte de la detección de incendios forestales, diversos proyectos han utilizado técnicas basadas en imágenes satelitales para detectar y monitorear incendios activos. Algunos de estos proyectos incluyen el uso de aprendizaje automático para mejorar la precisión y eficiencia en la detección de incendios, y a su vez en la toma de decisiones en tiempo real para la mitigación y prevención de desastres.

Uno de los proyectos antecesores al uso del Aprendizaje Automático en la detección de incendios se basa en el uso de imágenes satelitales geoestacionarias (Weaver, Lindsey, Bikos, Schmidt, & Prins, 2004, págs. 496-510) y el Algoritmo Automatizado de Quema de Biomasa de Incendios Forestales WF_ABBA (Wildfire Automated Biomass-Burning Algorithm) (Schmidt, Prins, & Feltz, 2002), que fue desarrollado para estudiar tendencias de quema de biomasa utilizando datos multiespectrales de los satélites ambientales geoestacionarios GOES (Geostationary Operational Environmental Satellite) y su uso en casos de estudio de incendios de junio de 2002.

Más adelante, después del lanzamiento del satélite Landsat-8 en el año 2013, el cual contiene un sensor generador de imágenes terrestres OLI (Operational Land Imager) (Knight & Kvaran, 2014), la cantidad de datos satelitales fue en aumento permitiendo el desarrollo de nuevos proyectos de monitoreo ambiental. Uno de esos proyectos (Schroeder, y otros, 2016) se encargó de desarrollar un algoritmo de detección de fuegos activos para usar con los datos diurnos y nocturnos proporcionados por OLI. El algoritmo se basa en el uso del canal de onda corta infrarrojo sensible al fuego, junto con los canales visibles e infrarrojo cercano. Además, se usa el análisis multitemporal para mejorar los resultados de la clasificación de pixeles.

Con el desarrollo del aprendizaje automático nuevas pesquisas se han apoyado en este campo, lo que ha permitiendo a los investigadores desarrollar proyectos de identificación de incendios precisos y eficientes (Ganapathi Subramanian & Crowley, 2018), (Langford, Kumar, & Hoffman, 2018), (Khryashchev & Larionov, 2020).

Tres proyectos que detallaremos a continuación son relevantes para esta investigación debido a los resultados obtenidos en su implementación. Uno de ellos realiza una comparativa de resultados en el proceso de identificación de incendios utilizando algoritmos con y sin el apoyo de DL, el segundo se enfoca en identificar el humo generado por las conflagraciones entre varios tipos de aerosoles, y el tercero presenta un modelo para desplegar algoritmos con redes neuronales en hardware embebido, el cual posee recursos computacionales y energéticos limitados.

4.1 Infrared-Image Processing for the DLR Firebird Mission.

El Centro Aeroespacial Alemán (DLR) opera la constelación FireBIRD (Briess, Jahn, & et.al., 2003), compuesta por dos misiones satelitales: Technology Test Platform (TET) y Bispectral Infrared Optical System (BIROS) (Lorenz, 2013). Esta constelación se dedica a la investigación científica de problemas ambientales, particularmente a la detección temprana de incendios desde el espacio. El enfoque de satélite y detector se basa en la tecnología DLR probada desarrollada durante la Misión Bispectral Infrad Detection (BIRD) que fue lanzada en 2001 y utilizada principalmente para la observación de incendios y actividad volcánica hasta 2004 (Halle, Fischer, Terzibaschian, Zell, & Reulke, 2019). Esta constelación utiliza canales espectrales en visible (VIS), infrarrojo cercano (NIR), onda media (MIR) y un canal de infrarrojo térmico (TIR). Una parte del proyecto se centra en el procesamiento de datos de imágenes TET y BIROS de FireBIRD. En la cadena de procesamiento estándar de FireBIRD, los productos de datos de nivel 1b y 2a se generan automáticamente para todos los usuarios luego de la recepción de datos en tierra.

Dos características de los sensores FireBIRD son únicas:

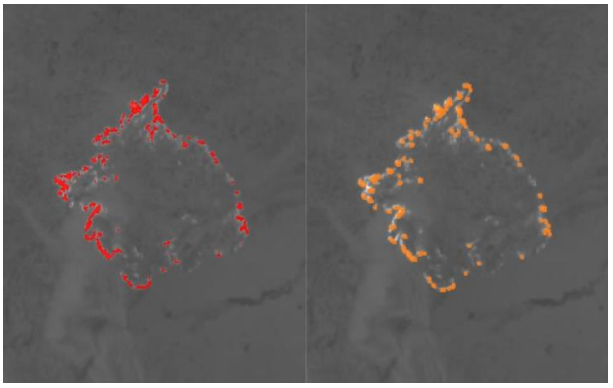
- La alta sensibilidad dinámica radiométrica para la evaluación cuantitativa de temperaturas normales y eventos de alta temperatura (HTE) en la misma escena.
- La evaluación del área efectiva del fuego en metros cuadrados, independientemente del número registrado de tamaños de grupo de fuego, que se da como el número de píxeles por grupo.

Para ciertos usuarios, como los bomberos, es esencial obtener información de incendios (ubicación y temperatura) de manera rápida una vez detectados. En estas aplicaciones, el procesamiento de datos debe hacerse directamente a bordo del satélite sin necesidad de una compleja cadena de procesamiento. Por ello, se evaluó un algoritmo alternativo de detección de incendios que emplea redes neuronales artificiales y se comparó con el procesamiento estándar FireBIRD de nivel 2.

Los resultados demostraron que el uso de pequeños satélites para monitorear el ambiente y detectar desastres naturales, especialmente incendios forestales, es muy efectivo. Por otro lado, la comparación entre el procesamiento estándar FireBIRD de nivel 2 con Redes Neuronales mostró resultados satisfactorios, demostrando que el aprendizaje profundo es una excelente opción para la detección de incendios (ver ilustración 3).

Figura 3.

Algoritmo FireBIRD izq. Algoritmo CNN der.



Nota: Briess, K.; Jahn, H.; et.al. B.: (2003). Fire recognition potential of the Bi-spectral InfraRed Detection (BIRD) satellite. *Int. J. Remote Sensing 24(4),* 865-872

4.2 Smokenet: Satellite Smoke Scene Detection Using Convolutional Neural Network With Spatial and Channel-Wise Attention.

La detección de humo a través de imágenes satelitales es una herramienta esencial para detectar y monitorear incendios forestales. Sin embargo, los métodos de detección de humo comúnmente usados se limitan a la discriminación entre seis clases (nubes, polvo, neblina, tierra, mar y humo), lo que reduce su utilidad en diferentes regiones de varios tipos (Ba, Chen, Yuan, Song, & Lo, 2019).

SmokeNet presenta una nueva referencia de detección de humo a partir de imágenes satelitales a gran escala, basada en datos del Espectro radiómetro de Imágenes de Resolución Moderada (MODIS) (Moderate Resolution Imaging Spectroradiometre (MODIS), sf). El dataset, compuesto por 6225 imágenes satelitales de diferentes áreas y regiones del mundo, se usó para establecer una línea de base para la detección de humo en imágenes satelitales. Se evaluaron varios modelos de clasificación de imágenes de última generación basados en aprendizaje profundo, así

como un nuevo modelo, SmokeNet, que incorpora la atención espacial para mejorar la representación de características en la clasificación de escenas de humo.

Los resultados de la prueba del método SmokeNet usando diferentes proporciones de imágenes de entrenamiento (16%, 32%, 48% y 64%) revelaron una mejor precisión y coeficiente Kappa (forma de medir cuán de acuerdo están varios evaluadores al evaluar una muestra). El modelo entrenado con 64% de imágenes de entrenamiento logró los mejores resultados, con una precisión del 92.75% y un coeficiente Kappa de 0.9130. Incluso con sólo el 16% de imágenes de entrenamiento, el modelo pudo mejorar la precisión de clasificación y el coeficiente de Kappa en al menos 4.99% y 0.06, respectivamente, en comparación con los modelos de última generación.

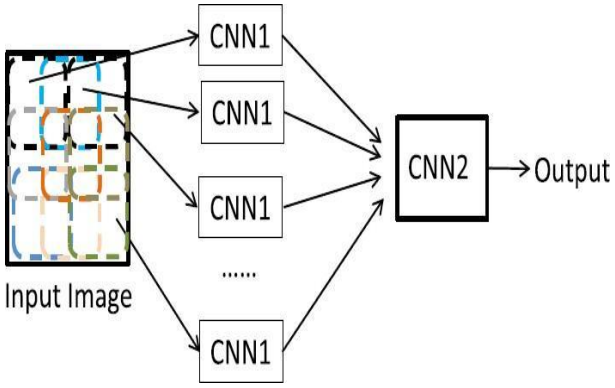
4.3 Lapped Convolutional Neural Networks for Embedded Systems.

La CNN ha logrado un gran progreso en varias aplicaciones de Inteligencia Artificial. No obstante, su complejidad es considerablemente alta y, por lo general, requiere un costoso despliegue de GPU (Unidad de Procesamiento Gráfico) (Graphic Processing Unit (GPU), sf) o FPGA (matriz de puertas lógicas programable en campo) (Field Programmable Gate Array (FPGA), sf), lo que no resulta rentable para muchos sistemas integrados. Por tanto, en este proyecto se ha diseñado una nueva arquitectura CNN, denominada CNN lapeada (LCNN), que es apropiada para sistemas embebidos de escasos recursos (Wang, Ng, & Liang, 2017).

La red está diseñada para descomponerse en dos o más etapas, cada una de las cuales se implementa mediante un módulo de hardware de baja resolución y baja complejidad. La imagen de entrada se divide en sub-imágenes del mismo tamaño con superposiciones diseñadas adecuadamente entre sí. Estas sub-imágenes se procesan secuencialmente por el módulo de hardware que implementa la primera etapa de la CNN. Las salidas de las diferentes sub-imágenes se fusionan y procesan en el siguiente módulo de hardware de bajo costo (ver ilustración 4).

Figura 4.

Redes Sobrepuestas.



Nota: Wang, X., Ng, H. W., & Liang, J. (2017, November). Lapped convolutional neural networks for embedded systems. *In 2017 IEEE Global Conference on Signal and Information Processing (GlobalSIP) (pp. 1135-1139). IEEE*

Al aplicar una CNN a mayor escala a toda la imagen con mayor resolución, se obtiene un resultado similar. Esto permite la reutilización de módulos de hardware de bajo costo para crear un sistema CNN económico y de mayor escala. La eficacia del método propuesto se ha demostrado mediante resultados experimentales. Esta solución es ideal para aquellas aplicaciones que requieren funcionalidades básicas de aprendizaje profundo, pero también deben cumplir con limitaciones en costos computacionales y consumo de energía.

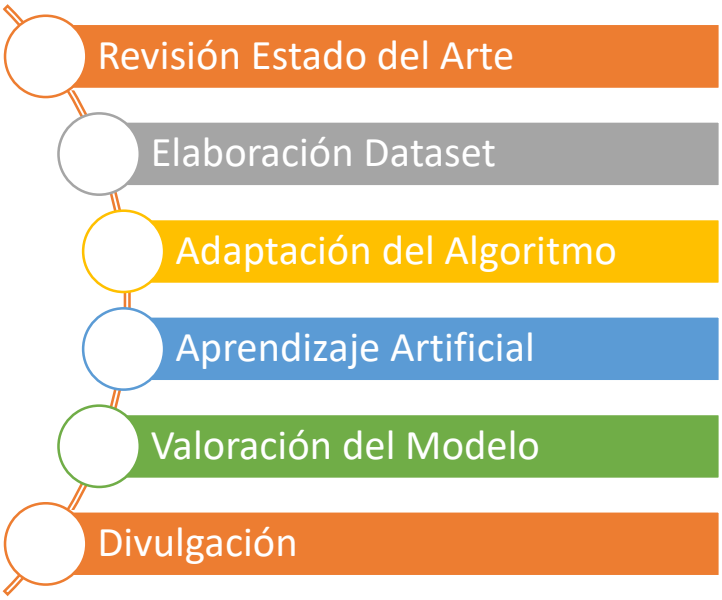
En base a lo presentado en el marco teórico y el estado del arte, se plantea una metodología para el desarrollo de la investigación que permita cumplir con los objetivos trazados.

5. Metodología

La metodología planteada para el desarrollo de este trabajo está basada en el ciclo de vida de IA (De Silva & Damminda, 2022) agrupando sus 19 etapas de desarrollo en 6 fases como se ve en la ilustración 5.

Figura 5.

Fases Metodológicas



Cada una de las fases se describe a continuación:

5.1 Revisión del Estado del Arte

Elaboración del estado del arte del tema de investigación a partir de la documentación existente. Hay que tener en cuenta que esta etapa es transversal y se ejecuta durante todo el tiempo de realización de este. Para su apreciación se puede remitir a la sección 3 del libro actual.

5.2 Elaboración del Dataset

Se llevó a cabo la creación de un conjunto de datos (Dataset) utilizando imágenes satelitales de acceso público. Primero se recopilaron las fechas y ubicaciones en las cuales se registraron

episodios de incendios. Posteriormente, en una plataforma en línea, se editaron las imágenes respectivas, las cuales fueron seleccionadas en función de un día específico y una zona geográfica determinada. Dichas imágenes fueron recortadas en las dimensiones definidas con anterioridad, con el fin de conformar el conjunto de datos. Los detalles específicos de este proceso se encuentran descritos en la sección 5 del presente documento. Algunas de las actividades que componen este paso son:

- Búsqueda en los portales digitales como DigitalGlobe (Digital Globe, sf), SpaceNet (Space Net, sf).
- Se contacta con instituciones científicas y académicas con experiencia en proyectos similares que permitan acceder a estas imágenes.
- Etiquetado del 80% de las imágenes que tienen evidencia de fuego para utilizarlas en el entrenamiento de la red neuronal. El otro 20% se utilizará en el proceso de validación.
- Selección del 80% de las imágenes sin evidencia de fuego para utilizarlas en el entrenamiento de la red neuronal. El otro 20% se utilizará en el proceso de validación.

5.3 Adaptación del Algoritmo

Selección del algoritmo de identificación de incendios más eficiente y confiable para implementarse sobre la placa de desarrollo integrada.

En base a la documentación existente revisada en la etapa de formulación del estado del arte, se adapta el algoritmo para el caso concreto de identificación de incendios en imágenes satelitales, el cual se puede encontrar con más detalle en la sección 6 del presente libro.

5.4 Aprendizaje Artificial

Aprendizaje de la red neuronal con el Dataset estructurado, el cual se puede observar con más detalle en la sección 7 del presente libro.

- Preparación del entorno de trabajo en el clúster de computación de alto rendimiento de la UIS (Wiki, s.f.).
- Entrenamiento de la red neuronal con el Dataset de imágenes satelitales, sobre la infraestructura de clúster de computación[1]. Este proceso no se ejecuta sobre el sistema embebido debido a la exigencia de recursos que implica.
- Elaboración de un procedimiento paso a paso para que pueda ser corroborado por otro usuario que desee replicar el experimento.

5.5 Valoración del Modelo

Luego de ejecutar el algoritmo entrenado sobre el sistema embebido, se realiza una evaluación del modelo, tomando en consideración las métricas de rendimiento de los resultados obtenidos. Para ver estos resultados con mayor detalle, consulte la sección 8 del presente libro.

5.6 Divulgación

Una vez evaluado el modelo por pares y publicado, se presentan los resultados y conclusiones obtenidos. Esta información podrá ser compartida de diversas formas, como una tesis de maestría, una ponencia o un artículo académico (Arguello, Hernández, & Ferreira, 2022).

Mediante la metodología expuesta, presentamos los resultados de la investigación. Cada una de las siguientes secciones muestra el logro de los objetivos planteados para el cumplimiento del trabajo.

[1] Se realiza con el apoyo del grupo de investigación CAGE

6. Dataset y Redes Neuronales

El Dataset fue creado con el objetivo de entrenar y hacer inferencia de redes neuronales. Está disponible en su repositorio correspondiente (Arguello, Hernández, & Ferreira, 2022), y los pasos seguidos para su elaboración y la selección de los modelos neuronales se explican detalladamente en los próximos párrafos.

6.1 Estructura del Dataset

Este proyecto tiene como objetivo identificar el fuego en imágenes satelitales de gran tamaño, utilizando un hardware embebido para realizar el análisis. Esto implica examinar una cantidad enorme de pixeles sin consumir muchos recursos computacionales y energéticos.

Entrenar redes convolucionales con imágenes de gran tamaño puede ser muy costoso en términos de tiempo y recursos. Esto se debe a que el tamaño de la imagen afecta directamente el número de parámetros que la red debe aprender. Cuanto mayor sea el tamaño de la imagen, más parámetros tendrá que aprender la red, lo que llevará más tiempo y recursos para entrenarla (Talebi & Milanfar, 2021), (Jin, Tanno, Mertzanidou, Panagiotaki, & Alexander, 2021).

Por otro lado, las investigaciones en donde se analiza la superficie terrestre se basa en el uso de imágenes de alta resolución (Maggiori, Tarabalka, Charpiat, & Alliez, 2017), debido a que utilizar una de menor tamaño no sería lo suficientemente detallada para analizar con precisión el área geoespacial. Esto se debe a que las imágenes satelitales de tamaño pequeño no ofrecen la misma resolución que las imágenes satelitales de mayor tamaño. Por lo tanto, no se utilizan imágenes pequeñas para el proceso de inferencia ya que no se obtendrían los detalles necesarios para realizar un análisis preciso.

Ante este dilema, la investigación se enfrenta al desafío de encontrar una solución que satisfaga los requerimientos mencionados. Para abordar esta situación, se crea un Dataset con imágenes pequeñas para el entrenamiento e imágenes grandes para la inferencia.

Del portal de acceso público de la NASA, llamado WORLDVIEW (National Aeronautics and Space Administration, 2020), se descargaron 1100 imágenes de la superficie terrestre a lo largo de una línea temporal con diferentes longitudes de onda. Estas imágenes fueron capturadas por el sensor óptico VIIRS (Cao, y otros, 2013), presente en los satélites NOAA-20 (Cao, y otros, 2018) y S-NPP (Weng, 2018). Las combinaciones de bandas espectrales obtenidas se identifican como "M3-I3-M11" (National Oceanic and Atmospheric Administration, 2017), cuyas características se pueden apreciar en la tabla 1.

Tabla 1.

Características Bandas Espectrales.

Bandas VIIRS			
Band	**Bandwidth (µm)**	**Band Explanation**	**Spatial Resolution**
M3	0.02	Visible/Reflective	750 m
M11	0.05	Shortwave IR	750 m
I1 (B)	0.08	Visible/Reflective	375 m
I2 (G)	0.039	Near IR	375 m
I3 (R)	0.06	Shortwave IR	375 m

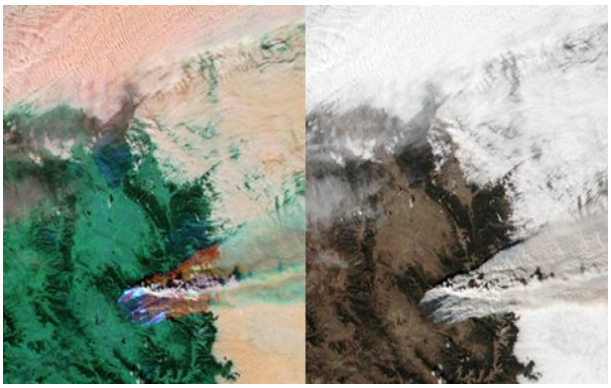
Nota: rammb. (sf). *VIIRS bands and bandwidths.* http://rammb.cira.colostate.edu/projects/npp/VIIRS_bands_and_bandwidths.pdf

Se seleccionaron las imágenes con esta combinación espectral debido al alto contraste que genera el color de la radiación térmica del fuego con respecto a la imagen en general y del humo desprendido por el fuego con respecto a las nubes y otros aerosoles como se observa en la

ilustración 6, lo que permite una identificación más eficaz con los algoritmos empleados en este proyecto.

Figura 6.

Bandas M3-I3-M11 izq. Bandas True Color der.



Nota: Worldview-NASA

Por otro lado, aunque el uso de imágenes “M3I3M11” en investigaciones no es nueva (Alonso & Millán, 2019), (Kadochnikov, 2019), (Sims, 2021), este es el primer estudio en el que se aplican para la identificación de fuego, lo que proporciona una valiosa alternativa para futuras investigaciones en este ámbito.

El conjunto de 1100 imágenes se dividió en dos paquetes. El primero, con 1000 imágenes, se usó para entrenar las redes neuronales; de estas, 500 contenían incendios forestales y 500 no. El segundo paquete, con 100 imágenes, se destinó al proceso de identificación con algoritmos, 50 de ellas con incendios forestales y 50 sin ellos. Estas imágenes presentan las siguientes características:

- Tamaño imágenes de entrenamiento: 224 x 224 pixeles
- Tamaño imágenes de identificación: 2048 x 2048 pixeles
- Resolución espacial: 125 metros
- Altitud: 200 km (Altitud promedio para micro satélites)

Para el entrenamiento de las redes neuronales se asignaron 800 imágenes (80%) para entrenamiento y 200 (20%) para validación. Debido al reducido número de imágenes y para aprovechar al máximo el entrenamiento de redes neuronales convolucionales, se aplicó una técnica llamada "Data Augmentation" (Zoph, y otros, 2019) al primer set de imágenes (800). Esta técnica permite aumentar el set de datos de entrenamiento aplicando transformaciones como Flip y Rotation a las imágenes originales, generando otras con los cambios reflejados. Como resultado, el conjunto de 800 imágenes ahora contiene 2230, sumando éste con el conjunto de validación de 200 imágenes, se obtiene un total de 2430 imágenes para el proceso completo de entrenamiento.

6.2 Redes Neuronales

Para identificar incendios forestales en imágenes satelitales, se seleccionaron dos modelos de CNN para su entrenamiento y evaluación: Inception V3 (Szegedy, Vanhoucke, Ioffe, Shlens, & Wojna, 2016) y Mobilenet V2 (Sandler, Howard, Zhu, Zhmoginov, & Chen, 2018). Estas selecciones se hicieron con el propósito de mantener un gasto mínimo de recursos computacionales sin sacrificar la efectividad del proceso de identificación.

Independientemente del modelo de red neuronal elegido para un proceso de identificación, un buen rendimiento en el entrenamiento requiere una gran cantidad de imágenes. Debido a la dificultad de construcción y al tiempo necesario para consolidar un extenso Dataset, es preciso recurrir a la técnica de aprendizaje por transferencia (Sarkar, Bali, & Ghosh, 2018), que aprovecha el conocimiento adquirido de modelos previamente entrenados para entrenar otros modelos, los cuales no necesariamente necesitan contar con una gran cantidad de imágenes. Por lo tanto, para este proyecto se debe realizar el aprendizaje por transferencia utilizando los pesos pre-entrenados de los dos modelos neuronales con el Dataset de imágenes ImageNET (Deng, y otros, 2009).

InceptionV3 se enfoca en reducir el consumo de recursos energéticos y computacionales, mejorando los modelos anteriores de inception. Las técnicas utilizadas para optimizar el rendimiento incluyen convoluciones factorizadas, regularización, reducción de dimensión y cálculos paralelizados.

Este modelo se construye utilizando la arquitectura InceptionV3 pre-entrenada en el conjunto de datos *ImageNet* (Shah, 2020). La red se adapta para clasificar imágenes en dos categorías (binaria). Primero, las imágenes se preprocesan utilizando el objeto *ImageDataGenerator* de la biblioteca *Keras*. Después, se define la estructura de la red neuronal utilizando la API funcional de *Keras*.

La entrada de la red es una imagen de 224x224 píxeles con 3 canales de color. El modelo InceptionV3 se utiliza como una capa base para extraer características, y se eliminan las capas superiores de clasificación para reemplazarlas por nuevas capas personalizadas. En particular, después de la capa base, se agrega una capa de agrupación global promedio, dos capas densas con 2048 y 1024 unidades de neuronas, respectivamente, y dos capas de abandono para evitar el sobreajuste del modelo. Finalmente, se agrega una capa de salida densa con dos neuronas y una función de activación *softmax* para producir la salida binaria.

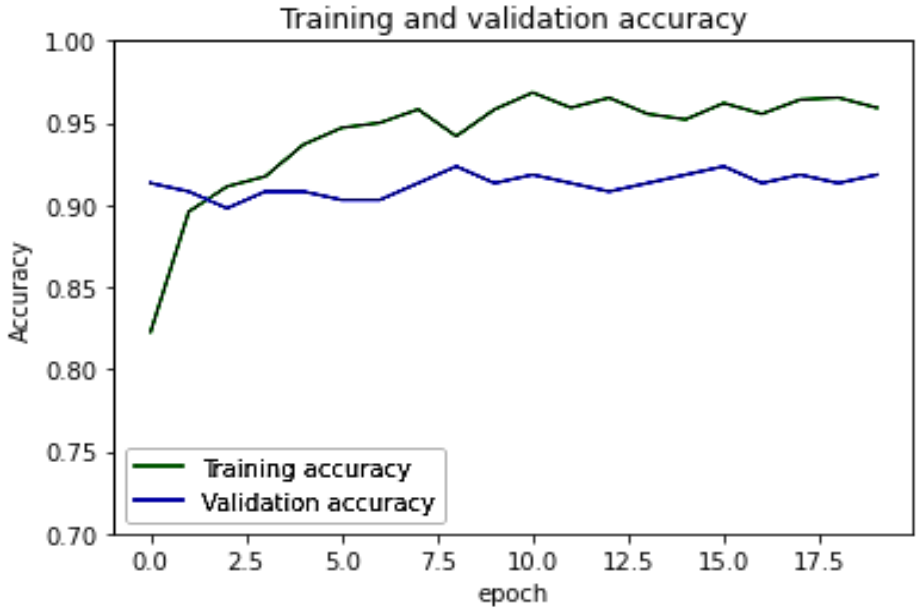
Después de compilar el modelo, se congela la base de la red (es decir, las primeras 249 capas) y se entrena solo las capas personalizadas. El optimizador utilizado es SGD con una tasa de aprendizaje de 0.0001 y un momento de 0.9.

El entrenamiento de la red se detiene después de 20 épocas o cuando la pérdida de validación y la pérdida de entrenamiento alcanzan un umbral de 0.1099. Por último, se muestra la precisión y la pérdida del entrenamiento y validación en un gráfico para facilitar la visualización.

Como resultado del proceso se obtiene una precisión de entrenamiento del 95.92, una precisión de validación del 91.84, una pérdida de entrenamiento de 0.1145 y una pérdida de validación del 0.2038 (ver ilustración 7 y 8).

Figura 7.

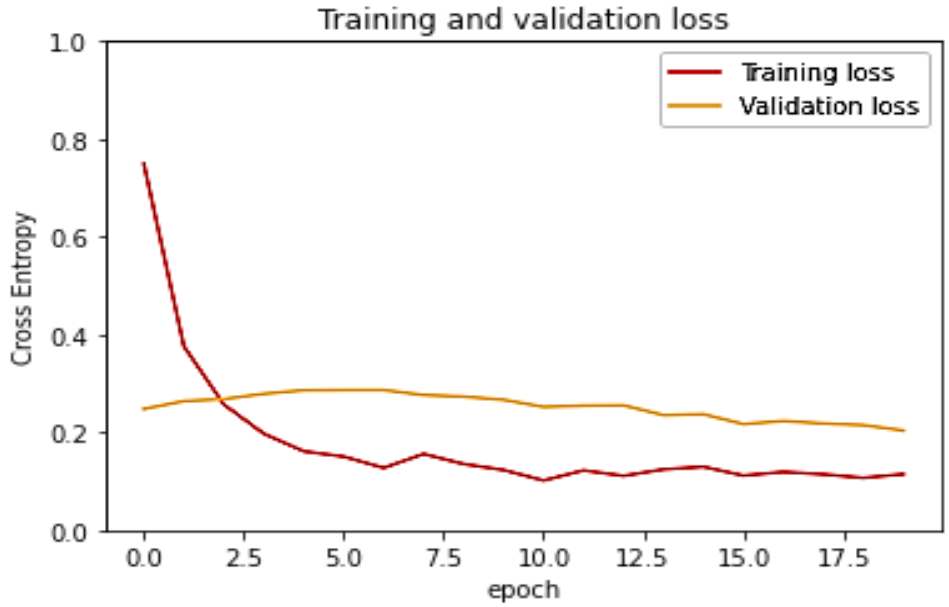
InceptionV3 Accuracy.



En la ilustración 7, el modelo InceptionV3 en entrenamiento muestra un aumento del 82.28 a 93.67 de precisión durante las primeras cinco (5) épocas y oscila entre el 94.69 y el 95.92 en las quince (15) épocas restantes. El proceso de validación mantuvo el nivel de precisión dentro de la franja de los 89.80 y el 91.84

Figura 8.

InceptionV3 Loss.



La ilustración 8 muestra que el modelo inceptionV3 durante el entrenamiento presenta un nivel de pérdida descendente desde 0.7488 hasta 0.1274 durante las primeras siete (7) épocas, luego oscila entre 0.1015 y 0.1557. En el proceso de validación, el nivel de pérdida siempre fluctúa entre 0.2038 y 0.2865.

MobilenetV2 está diseñado para su uso en dispositivos móviles, lo que significa que requiere un consumo computacional y energético bajo. Conserva muchas de las características de su predecesor MobilenetV1, y añade dos mejoras: cuellos de botella lineales entre capas y conexiones de atajo entre los cuellos de botella.

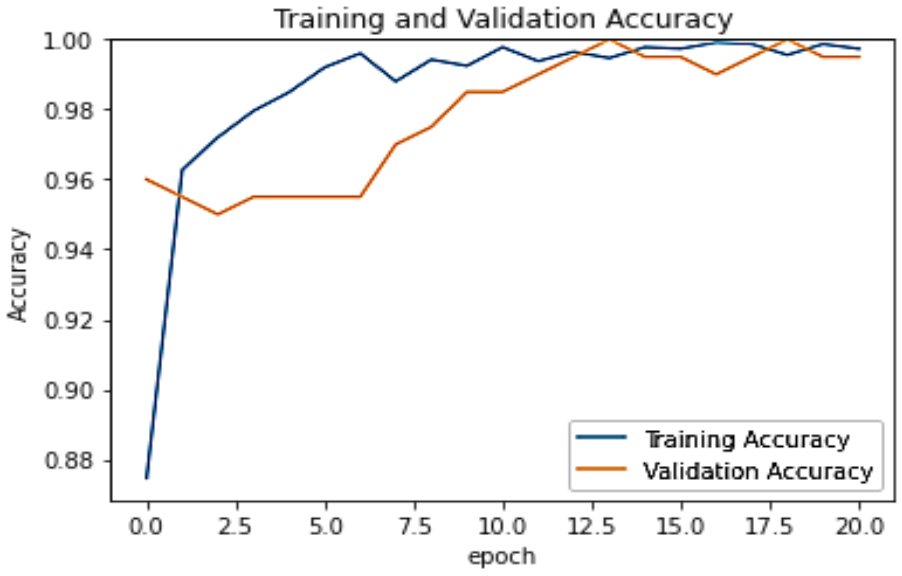
En esta investigación, la red neuronal utilizada por Sahoo (2021) es una red pre-entrenada MobileNetV2, la cual se utiliza como base del modelo. Se realiza un *fine-tuning* de las últimas capas de la red, para ajustar los pesos a nuestro problema específico. Para ello, se agregan una capa *GlobalAveragePooling2D*, que toma los mapas de características de la red y calcula la media

global para cada canal de características, reduciendo la dimensionalidad de la salida. Luego se agrega una capa Dense con 1 unidad de salida, ya que es un problema binario. La función de activación por defecto de esta capa es lineal, ya que se utiliza una capa previa de *GlobalAveragePooling2D*, por lo que se utilizan logits para la clasificación. El modelo es compilado con la función de pérdida *BinaryCrossentropy* y el optimizador *RMSprop*. Finalmente, se entrena el modelo primero con 20 épocas, congelando las capas de la red pre-entrenada, y luego con 20 épocas adicionales, desbloqueando las últimas capas de la red para realizar un *fine-tuning*.

Como resultado del proceso se obtiene una precisión de entrenamiento del 99.73, una precisión de validación del 99.50, una pérdida de entrenamiento de 0.0096 y una pérdida de validación del 0.0110 (ver ilustración 9 y 10).

Figura 9.

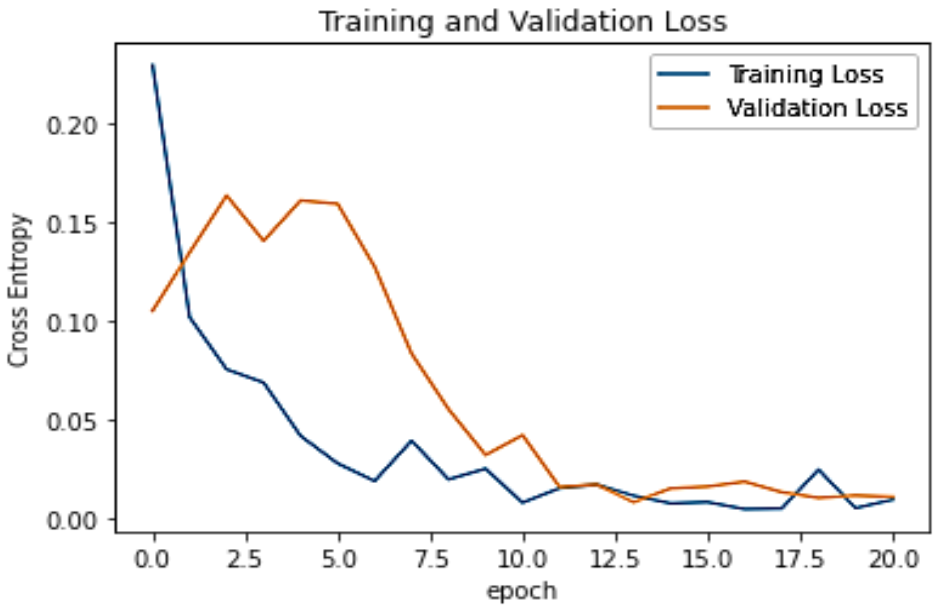
MobilenetV2 Accuracy.



En la ilustración 9, el modelo mobilenetV2 durante su entrenamiento presenta un ascenso rectilíneo de 87.47 a 97.20 de precisión después de solo dos (2) épocas, y luego continúa con una parábola ascendente poco profunda hasta alcanzar 99.73 durante las dieciocho (18) épocas siguientes. En el proceso de validación, el nivel de precisión asciende desde 95 hasta 99.50.

Figura 10.

MobilenetV2 Loss.



La ilustración 10 muestra que el modelo mobilenet V2 durante el entrenamiento presenta un nivel de pérdida de curva descendente desde 0.2294 a 0.0190 durante las primeras seis (6) épocas, para luego marcar una línea fluctuante entre 0.0081 y 0.0394. En el proceso de validación, el nivel de pérdida dibuja una curva que comienza en 0.1052, sube a 0.1636 durante las primeras cuatro (4) épocas, luego desciende y se mantiene entre 0.322 y 0.110.

Los modelos presentados cuentan con una alta precisión inicial gracias al aprendizaje por transferencia, que utiliza los pesos previos obtenidos del modelo pre entrenado. Esto se refleja en los resultados de entrenamiento y validación, donde el modelo mobilenetV2 presenta porcentajes más elevados que el modelo inceptionV3.

7. IGNIS

Durante el análisis del marco teórico, se observó la necesidad de utilizar una gran cantidad de imágenes para el entrenamiento y validación de algoritmos basados en redes neuronales convolucionales (CNN), así como la necesidad de tecnologías de alto rendimiento computacional para el procesamiento. Aunque este procedimiento se llevó a cabo en el proyecto, también se planteó una alternativa que simplifica la identificación de incendios en imágenes satelitales, y se despliega en sistemas computacionales embebidos, sin sacrificar eficiencia, rendimiento y efectividad.

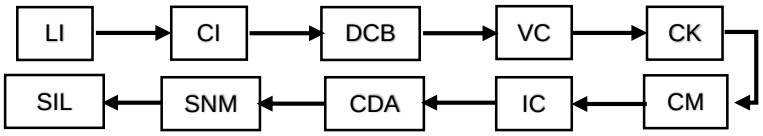
Se presenta el algoritmo IGNIS como esa alternativa viable para la detección de incendios en imágenes satelitales. Lo que lo hace especialmente atractivo es que no requiere un Dataset a priori ni un entrenamiento previo para su despliegue, además de que su consumo de recursos energéticos y computacionales (RAM, GPU, CPU) es reducido, lo que resulta acorde con el objetivo del proyecto.

El algoritmo IGNIS se enfoca en la identificación de incendios mediante el uso de técnicas de visión artificial como el filtrado basado en el color y la selección de pixeles por diferencia con el contorno, todo esto con el apoyo de la biblioteca OpenCV (Open Computer Vision,, sf). Sin embargo, a diferencia de otros proyectos que también usan estas técnicas (Hatekar, Manwani, Patil, & Parekh, 2017), (Trambitckii, Anding, Musalimov, & Linss, 2015), este algoritmo no está diseñado para ser utilizado con videos o imágenes con color real de cámaras de vigilancia. Además, puede ser ejecutado en un dispositivo embebido y en una computadora tradicional.

IGNIS presenta una serie de pasos como lo muestra la ilustración 11, desde la lectura de las imágenes hasta la identificación de los incendios, entregando las coordenadas y el trazo de los cuadros delimitadores.

Figura 11.

Secuencia de IGNIS.

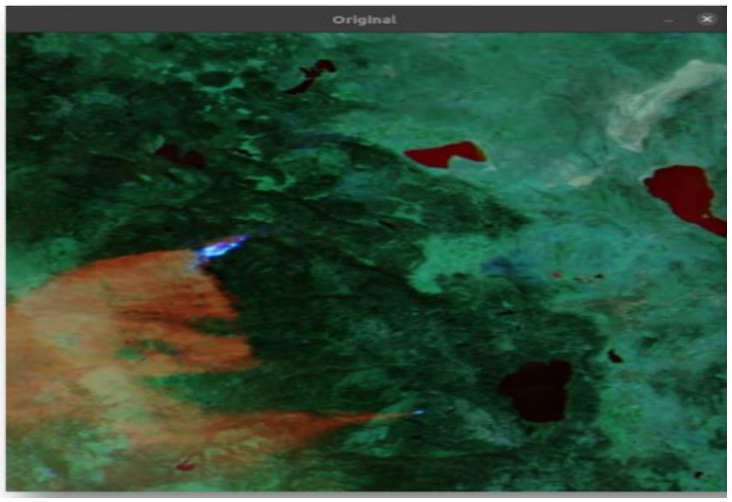


Los pasos que conforman este algoritmo se presentan a continuación:

- Lectura de las imágenes (LI): Estas imágenes se leen de una carpeta con una ruta específica al interior del algoritmo.

Figura 12.

Incendio-Original



- Conversión de imágenes (CI): Las imágenes se convierten de RGB a HSV. La representación de colores HSV (*Hue, Saturation, Value*) separa la información de color de la información de brillo y permite una mejor separación de los objetos en la imagen.

- Definición del color buscado (DCB): se establecen los rangos de matiz, saturación y brillo correspondientes al color que se debe identificar al interior de las imágenes. Estos rangos se presentan como 6 valores, dos para cada componente (H, S, V) pertenecientes al valor mínimo y máximo. Para el caso particular de este proyecto, en el tipo de imágenes satelitales elegidas, los 6 valores correspondientes al color azul que representa el fuego son: hMin = 100, hMax = 140, sMin = 100, sMax = 255, vMin = 100, vMax = 255. Estos valores se establecen manualmente a través de un script, el cual crea una ventana con varias barras deslizantes (*trackbars*) que permiten al usuario ajustar los valores mínimos y máximos de los componentes H, S y V de la imagen para detectar el objeto deseado. Luego, se leen los valores actuales de las barras deslizantes y se utilizan para definir un rango de colores. También se leen los valores de las barras deslizantes para definir el tamaño del kernel utilizado en las operaciones de morfología matemática.

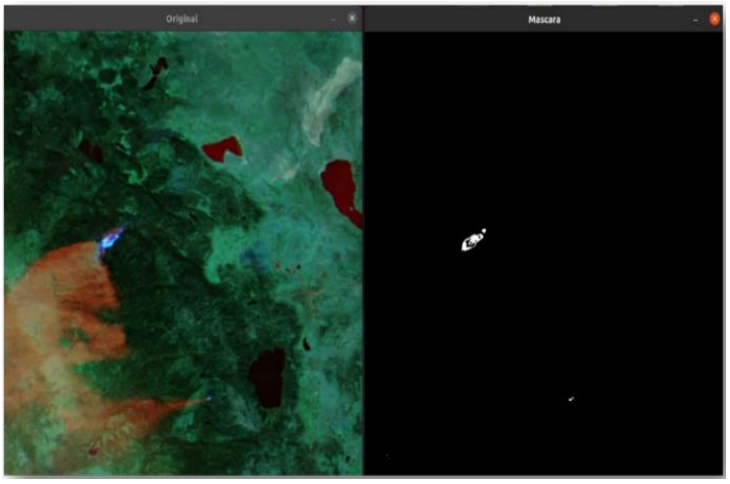
- Vectores de color (VC): Se crean dos vectores del color seleccionado, un vector para valores mínimos y otro para valores máximos.

- Creación del kernel (CK): Se define y crea un kernel que se utilizará para la eliminación de ruido en el proceso de identificación. Este kernel es una matriz de unos (1) necesaria en la limpieza de las imágenes.

- Creación máscara (CM): Se crea una máscara limpia sobre las imágenes, que identifica los elementos del color particular utilizando el kernel y los arrays de color definidos anticipadamente.

Figura 13.

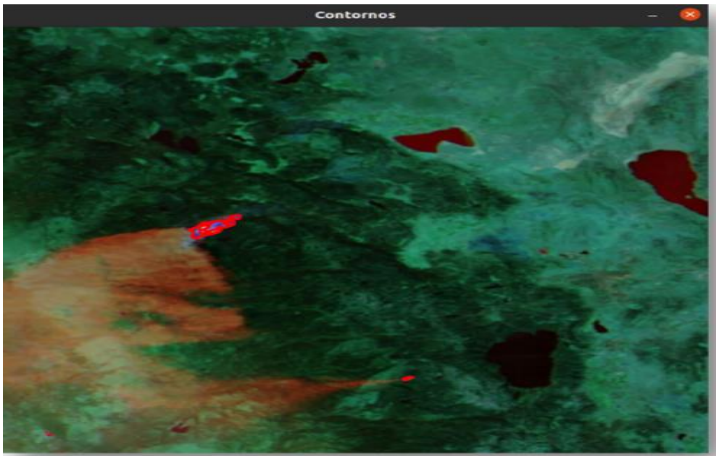
Incendio-Máscara.



- Identificación de contornos (IC): Estos contornos se identifican sobre las máscaras y se dibujan sobre las imágenes.

Figura 14.

Incendio-Contornos.

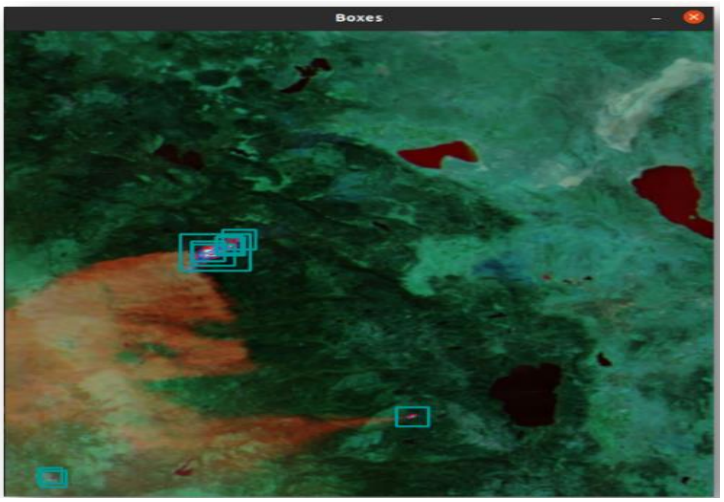


- Cuadros delimitadores y áreas (CDA): Si no se encuentran contornos en la máscara se inserta uno (1) en un vector que contiene el listado de clasificación y se lee la siguiente imagen, de lo contrario, si se encuentran contornos en la máscara, se inserta cero (0) en el listado de clasificación, se calcula el área de cada contorno y se genera su cuadro delimitador respectivo. El

área de cada contorno se guarda en un vector, así como las coordenadas de su cuadro delimitador en otro vector.

Figura 15.

Incendio-Cuadros.

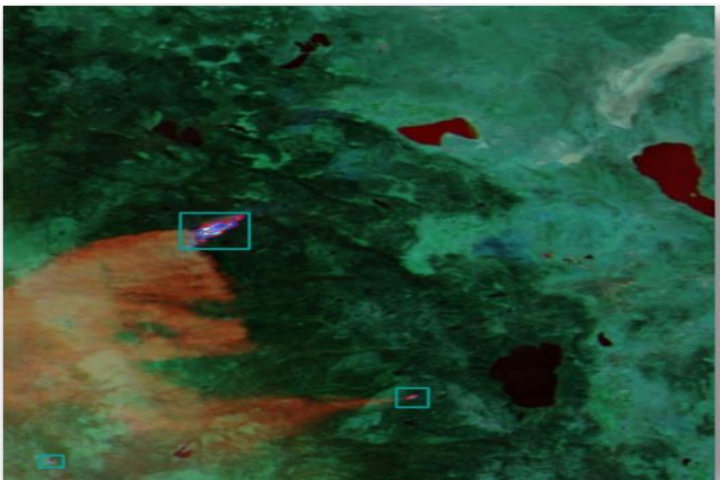


- Supresión no máxima (SNM): Al igual que en el proceso de identificación utilizando redes neuronales convolucionales CNN, se genera un exceso de cuadros delimitadores para identificar a un mismo elemento, lo que conlleva a la necesidad de utilizar el algoritmo de supresión no máxima (NMS) para seleccionar solo aquellos que cumplen los umbrales o puntajes definidos. Cuando se utilizan CNN, estas entregan a cada cuadro delimitador un puntaje de confiabilidad, pero como en este caso los cuadros delimitadores no provienen de una CNN, lo que se utiliza como puntaje de confiabilidad es el área del contorno. Para esto, la función *cv2.dnn.NMSBoxes()* toma como entrada una lista de coordenadas, que representan las cajas delimitadoras de fuegos identificados en una imagen, y aplica NMS para suprimir cajas superpuestas y retener solo las más seguras. La función también toma parámetros adicionales, incluyendo el parámetro "boundarea", que representa el área total de la imagen o cuadro, y el *score_threshold* y el parámetro *nms_threshold*, que representan umbrales para la puntuación de la

confianza y la superposición de unión de intersección (*IoU*), respectivamente. Las cajas con puntuaciones inferiores al *score_threshold* se descartan, mientras que las cajas con superposiciones *IoU* por encima del *nms_threshold* se suprimen. Estos criterios o umbrales para la SNM se definen experimentalmente, ajustándolos en función de los resultados obtenidos durante la evaluación del algoritmo en un conjunto de datos de prueba. El objetivo es encontrar un equilibrio entre la sensibilidad de identificación y la precisión de eliminación de detecciones redundantes.

Figura 16.

Incendio-Supresión.



- Salvar imágenes y listados (SIL): Ahora se dibujan en las imágenes los cuadros delimitadores seleccionados en el paso anterior y se guardan los cambios. Finalmente se crean archivos csv para los vectores de coordenadas y para el listado de clasificación.

La visión artificial es una herramienta poderosa que permite abordar una gran variedad de problemas a través del tratamiento de imágenes. El algoritmo IGNIS es un ejemplo de cómo esta tecnología puede ser aplicada en diversos contextos. Además, el modelo se puede replicar

fácilmente por otros usuarios, lo que permite una diseminación favorable del conocimiento y una aplicación satisfactoria del mismo.

Aunque el objetivo declarado del proyecto de investigación es el desarrollo de modelos para la clasificación de imágenes, se considera apropiado que el algoritmo IGNIS desempeñe además una función de detección en el proceso de la misma. Al adicionar la detección a la clasificación, IGNIS brinda una mejora adicional en la capacidad de identificar incendios de manera precisa. Además de detectar áreas de fuego, IGNIS es capaz de clasificarlas como incendios, lo que permite una respuesta más precisa y eficiente ante situaciones de emergencia. La combinación de detección y clasificación en un solo algoritmo mejora la confiabilidad de los resultados y facilita la toma de decisiones.

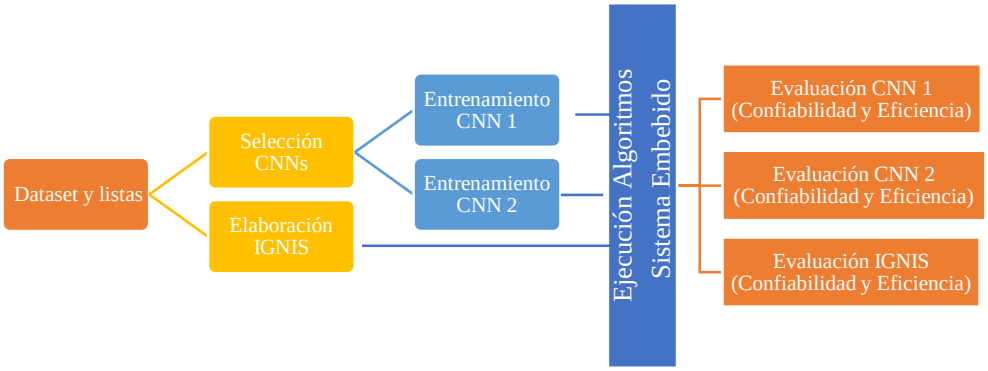
Sin embargo, el uso del algoritmo IGNIS no es apropiado en situaciones donde los objetos a ser identificados presenten una variedad de colores, tales como animales o vehículos, o cuando el color de los objetos no genera un contraste significativo con el fondo de la imagen.

8. Workflow

En el contexto del proyecto de identificación de incendios, se estableció un flujo de trabajo compuesto por una serie de actividades relacionadas entre sí. Al repetirse, estas acciones conducen a resultados similares a los descritos en este documento. Como lo muestra la ilustración 17, en algunos momentos, estas tareas pueden realizarse de manera simultánea, pero en otras, deben ser ejecutadas de forma secuencial debido a la interdependencia entre ellas.

Figura 17.

Workflow.



Para una mayor comprensión de las tareas que conforman este flujo de trabajo, se ofrece una descripción detallada de cada una de ellas.

- Dataset y listas: Esta actividad consiste en la creación de un Dataset de imágenes para entrenamiento y ejecución. Además, se elabora una lista de valores reales a partir de las imágenes destinadas para la inferencia, que servirá como insumo para evaluar la confiabilidad de los resultados de los algoritmos.

- Selección CNNs: Seleccionar modelos neuronales que sean apropiados para el proyecto, teniendo en cuenta sus características.

- Elaboración IGNIS: Este algoritmo de visión artificial se construyó durante el desarrollo del proyecto, y se encuentra disponible en el repositorio.

- Entrenamiento CNNs: Se realiza el entrenamiento de las redes neuronales sobre un hardware diferente a la tarjeta embebida que se utilizará para la inferencia. Este hardware debe poseer gran capacidad de cómputo y memoria, puesto que el proceso de convoluciones lo requiere.

- Ejecución de Algoritmos – Sistema Embebido: Después del entrenamiento de las redes neuronales convolucionales (CNNs), se utilizan los archivos de soporte para inferencia (h5) en los

algoritmos de identificación. Para ello, es necesario ejecutar un script en segundo plano que tome mediciones de consumo de la tarjeta NVIDIA Jetson Nano (Nvidia, s.f.). Este script, genera como resultado una lista con las predicciones de fuego en las imágenes y un archivo de texto con las lecturas de la tarjeta electrónica.

- Evaluaciones: Utilizando el archivo de texto con las lecturas de consumo de recursos de la tarjeta, un script se encarga de organizar y convertir la información a formato CSV para posteriormente ser graficada y utilizada como insumo para calcular las métricas de eficiencia. Por otro lado, la lista con las predicciones se compara con la lista original, generando como resultado la matriz de confusión y a su vez las métricas de confiabilidad.

Para una mejor comprensión del proceso de trabajo, se recomienda revisar el repositorio de GitHub (Github, sf), donde se ofrecen detalles sobre cada una de las actividades y los archivos principales mencionados previamente.

Una vez explicado el procedimiento para replicar este proyecto de investigación, se presentan los resultados alcanzados y las conclusiones derivadas del uso de los algoritmos de identificación.

9. Métricas y Evaluación

En el ámbito científico, es esencial evaluar las soluciones algorítmicas propuestas para abordar una problemática específica mediante el uso de medidas y la recolección de datos durante la ejecución del algoritmo, con el objetivo de calcular las métricas establecidas. En este proyecto, se ha establecido la importancia de medir tanto la confiabilidad como la eficiencia de la propuesta algorítmica.

9.1 Métricas

Para determinar la confiabilidad, se comparan las predicciones generadas por los algoritmos durante su ejecución con una lista previamente establecida de puntuaciones reales. Para medir la eficiencia, se debe ejecutar los algoritmos en un sistema embebido. La tarjeta Jetson Nano de Nvidia ofrece una herramienta llamada Tegrastats para medir los diferentes consumos durante un periodo de tiempo determinado.

9.1.1 Confiabilidad

La confiabilidad dentro del proceso de evaluación nos permite determinar el nivel de consistencia del modelo a través de consecutivos eventos de medición. Para esto, se utiliza la Matriz de Confusión, que nos muestra el resultado de la comparación entre los resultados ciertos y los obtenidos luego de la ejecución del algoritmo. La Matriz de Confusión se genera para cada algoritmo a partir de las listas de predicciones.

La tabla 2 muestra los resultados posibles al comparar las respuestas esperadas con las generadas por los algoritmos, los cuales se dividen en cuatro variables.

Tabla 2.

Resultados Matriz de Confusión

REAL/GENERADO	POSITIVO	NEGATIVO
POSITIVO	VP	FN
NEGATIVO	FP	VN

Donde:

- VP - verdadero positivo: cuando realmente es positivo y se genera positivo.
- FN - falso negativo: cuando realmente es positivo y se genera negativo.
- FP - falso positivo: cuando realmente es negativo y se genera positivo.

- VN - verdadero negativo: cuando realmente es negativo y se genera negativo.

A partir de estas variables se construyen las métricas que permitirán evaluar la fiabilidad del algoritmo.

- Precisión (Precision): Es el índice resultante entre los verdaderos positivos y el total de positivos generados por el algoritmo.

$$\mathrm{VP}/(\mathrm{VP}+\mathrm{FP})$$

- Exactitud (Accuracy): Es el índice resultante entre los aciertos, tanto positivos como negativos, y el total de resultados generados por el algoritmo.

$$\mathrm{VP}+\mathrm{VN}/(\mathrm{VP}+\mathrm{FN}+\mathrm{FP}+\mathrm{VN})$$

- Sensibilidad (Recall): Es el índice resultante entre los verdaderos positivo y el total de positivos reales.

$$\mathrm{VP}/(\mathrm{VP}+\mathrm{FN})$$

- F1-Score: Permite cotejar dos métricas en una sola, sensibilidad y precisión, y es muy utilizada cuando las clases dentro del dataset son desiguales.

$$2*(\mathrm{Sensibilidad}*\mathrm{Precisión})/(\mathrm{Sensibilidad}+\mathrm{Precisión})$$

9.1.2 Eficiencia

La eficiencia algorítmica es un enfoque crucial para el desarrollo de modelos de aprendizaje automático, dado que requieren una gran cantidad de recursos computacionales y energéticos. Por esta razón, es importante evaluar el consumo de estos recursos y tratar de reducirlo. Esta preocupación es aún mayor en sistemas embebidos o arquitecturas integradas, debido a sus limitaciones (Hernandez, Montenegro, & Navaux, 2016).

Las métricas a tener en cuenta con los modelos del proyecto son:

- Consumo Energético: Medido en miliwatts (mw) y corresponde al consumo promedio de la tarjeta embebida durante la prueba con cada algoritmo
- Consumo de memoria RAM: Medido en megabytes (MB). La tarjeta Jetson Nano comparte la memoria RAM con la GPU, es decir, no se encuentra separada.
- Temperatura del Dispositivo: Medida en grados centígrados (°C) y corresponde a la temperatura que registra toda la tarjeta durante el tiempo transcurrido.
- Consumo de GPU: Medido en porcentaje (%) y corresponde al uso que se da de la GPU en su procesamiento.

Se creó un script para tomar mediciones, el cual inicia un algoritmo en segundo plano para recopilar datos de consumo de la tarjeta Jetson Nano. Como ejemplo, se presenta la ejecución de IGNIS como se muestra en la ilustración 18.

Figura 18.

Script de Medición.

```
#!/usr/bin/env python3
import time
import subprocess
import threading

def funcion():
    p=subprocess.run(["python3","ignis.py","&"], stdout=subprocess.PIPE)

hilo=threading.Thread(target=funcion)

tegrastats_cmd = f"sudo tegrastats --interval 1000"
cmd = f"{{ echo $(date -u) & {tegrastats_cmd}; }} > datos_ignis.txt"
lecturas = subprocess.Popen(cmd, shell=True)

i=1

while True:

    if i==1:
       time.sleep(5)
       hilo.start()
       print("el algoritmo ya empezÃ³")
    if i > 1 & hilo.is_alive() == False:
       print("ya terminÃ³ el algoritmo")
       time.sleep(5)
       lecturas.kill()
       break
    i=i+1
```

Este algoritmo crea un hilo (thread) que ejecuta la función "funcion()" en segundo plano, mientras en el hilo principal se ejecuta un comando de forma continua "sudo tegrastats --interval 1000" y se guarda la salida en un archivo.

La función "funcion()" utiliza la biblioteca subprocess para ejecutar otro script llamado "ignis.py" en paralelo utilizando el operador "&" para indicar que se ejecute en background.

El código crea un objeto "hilo" de la clase threading.Thread y asigna la función "funcion()" como el objetivo o método a ejecutar. Luego se crea un objeto "lecturas" usando la función Popen de la biblioteca subprocess, el cual ejecuta el comando "sudo tegrastats --interval 1000" y redirige la salida a un archivo llamado "datos_ignis.txt".

En un bucle infinito se verifica si es la primera iteración, en caso afirmativo se duerme el hilo principal por 5 segundos y se inicia el hilo que ejecuta la función "funcion()". Si la iteración es mayor a 1 y el hilo creado no está activo se imprime "ya terminó el algoritmo", el hilo principal se duerme por 5 segundos y finaliza el objeto "lecturas" y se sale del bucle. Cada iteración i se incrementa en 1.

Este algoritmo permite ejecutar dos tareas en paralelo, una en background y otra en foreground y se puede utilizar para monitorear el rendimiento de la CPU, GPU, temperatura, entre otros recursos mientras se ejecuta otro script.

9.2 Evaluación

La evaluación de los algoritmos de inteligencia artificial es un paso crucial para garantizar que el proyecto sea exitoso. Esta evaluación permite medir el rendimiento de los algoritmos, identificar problemas de rendimiento y asegurar que estos cumplan con los objetivos establecidos.

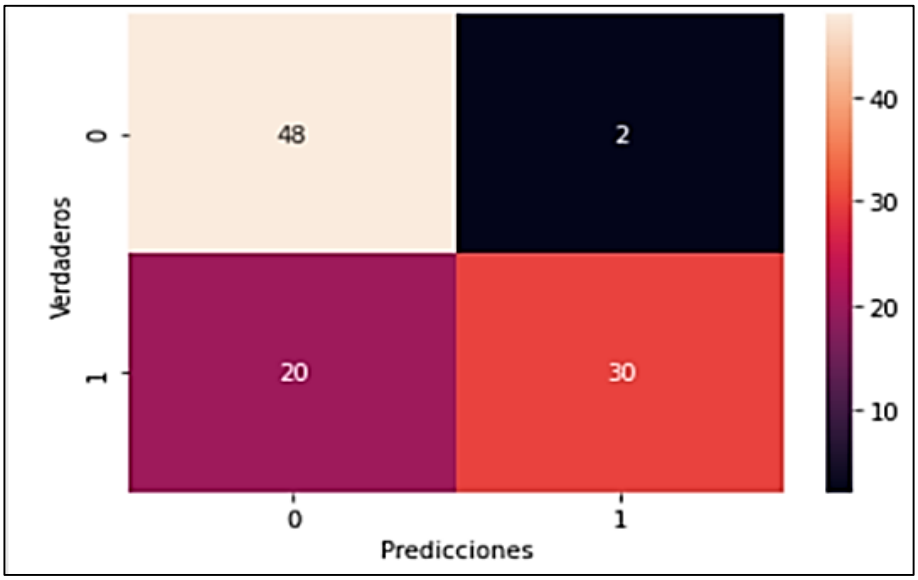
9.2.1 Confiabilidad

Haciendo uso de las listas de puntuaciones reales y las predichas, se crea para cada algoritmo la matriz de confusión. Esta matriz proporciona una visión general de cómo los modelos clasifican los datos de entrada. Muestra cuántos de los datos de entrada se clasificaron correctamente y cuántos se clasificaron incorrectamente. Esta información es esencial para evaluar el desempeño del modelo de clasificación y mejorarlo.

Las ilustraciones 19, 20 y 21 muestran las matrices de confusión de los algoritmos InceptionV3, MobilenetV2 e IGNIS respectivamente.

Figura 19.

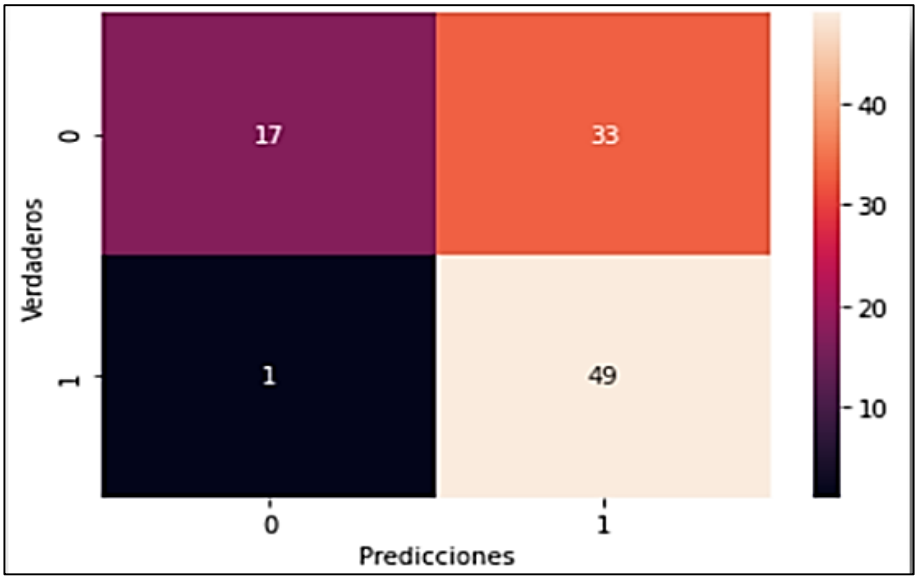
Confusión InceptionV3.



La ilustración 19 revela que el algoritmo InceptionV3 es más preciso para identificar imágenes con fuego, pero más propenso a errores con aquellas que no lo tienen.

Figura 20.

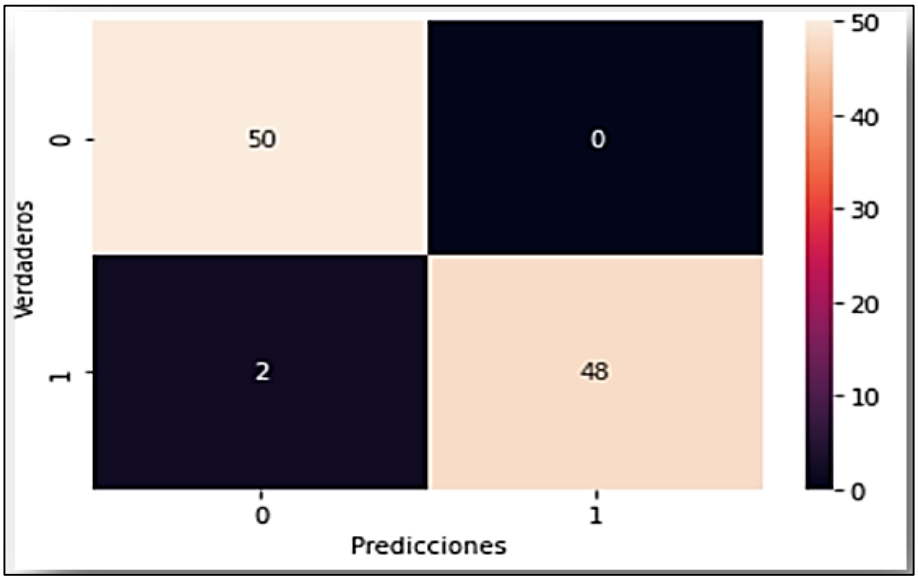
Confusión Mobilenet



La ilustración 20 revela que el algoritmo mobilenetV2 es más confiable para identificar imágenes sin fuego y más propenso a cometer errores con aquellas que contienen fuego.

Figura 21.

Confusión IGNIS.



La ilustración 21 revela que el algoritmo IGNIS es confiable en la identificación de imágenes con fuego y sin fuego, demostrando que es eficaz a pesar del error numéricamente bajo en el segundo caso.

Con la matriz de confusión de cada algoritmo, se obtuvieron sus respectivas métricas de confiabilidad, las cuales se presentan en la tabla 3.

Tabla 3.

Resultados Confiabilidad

Algoritmos	Precisión	Recall	Accuracy	F1-Score	Avg
IGNIS	98%	98%	98%	98%	98%
InceptionV3	82%	78%	78%	77%%	79%
MobilenetV2	77%	66%	66%	62%	68%

La tabla muestra que el algoritmo IGNIS es mejor en términos de precisión, recall, accuracy y F1-Score. Esto significa que es el más preciso, el que mejor recuerda los detalles y el más exacto entre los tres algoritmos.

Los resultados obtenidos de las inferencias con las redes neuronales son inferiores a las puntuaciones logradas en sus entrenamientos (>=90%). Es necesario indagar en las posibles causas de esta puntuación y considerar alternativas para mejorar el rendimiento de los algoritmos en esta tarea específica.

Una posible explicación para los resultados de los algoritmos InceptionV3 y MobilenetV2 podría derivar de analizar la diferencia en tamaño entre las imágenes utilizadas en el entrenamiento y la inferencia. El algoritmo InceptionV3 fue entrenado con imágenes de 214 X 214 pixeles, mientras que el algoritmo MobilenetV2 reduce internamente el tamaño de las imágenes a 160 X 160 pixeles. Por otro lado, las imágenes de evaluación son de 2048 X 2048 pixeles. Esta diferencia en tamaño podría explicar por qué el algoritmo MobilenetV2 generó resultados más bajos, ya que

el tamaño de las imágenes de entrenamiento es aproximadamente 13 veces menor que el tamaño de las imágenes de evaluación. Mientras que en el caso de InceptionV3 la diferencia es de aproximadamente 10 veces. Esto sugiere que el tamaño de las imágenes utilizadas para entrenar y evaluar los algoritmos podría ser un factor crítico en el rendimiento final.

Sin embargo, la diferencia en el tamaño de las imágenes también plantea una dificultad adicional en el desempeño de los algoritmos. Al reducir el tamaño de las imágenes de inferencia al interior de los algoritmos, también se minimiza el área ocupada por el fuego dentro de la misma, lo que en algunos casos puede llegar a ser menor al 1% del tamaño total de la imagen. Esto podría afectar significativamente la capacidad de los algoritmos para detectar y clasificar correctamente el fuego, ya que un menor tamaño del área de fuego en las imágenes de evaluación podría generar una menor precisión y exactitud.

En cambio, el algoritmo IGNIS no presenta este problema, ya que no se basa en el entrenamiento de redes neuronales, sino que analiza directamente el color de los pixeles para identificar los que corresponden al fuego. A diferencia de los algoritmos anteriormente mencionados, el tamaño de la imagen solo afecta al tiempo de ejecución del algoritmo, ya que cuanto mayor sea el tamaño de la imagen, mayor será el espacio a analizar y, por lo tanto, mayor será el tiempo requerido, pero no la capacidad de revisar los pixeles para su clasificación.

En conclusión, se ha presentado un análisis relativo entre los algoritmos IGNIS, InceptionV3 y MobilenetV2 en términos de las métricas de confiabilidad. Los resultados obtenidos muestran que el algoritmo IGNIS tiene un desempeño superior en comparación con los otros dos algoritmos evaluados. Se ha sugerido que esta diferencia en el rendimiento podría ser atribuible a la diferencia en el tamaño de las imágenes utilizadas en el entrenamiento y la inferencia.

Es importante destacar que, al reducir el tamaño de las imágenes, también se minimiza el área ocupada por el fuego, lo que puede afectar significativamente la capacidad de los algoritmos para identificar y clasificar correctamente el fuego.

Sin embargo, para el algoritmo IGNIS, que no se basa en el entrenamiento de redes neuronales, el tamaño de la imagen solo afecta al tiempo de ejecución. En general, se puede concluir que el algoritmo IGNIS es una opción confiable para la identificación de incendios en imágenes satelitales de gran tamaño.

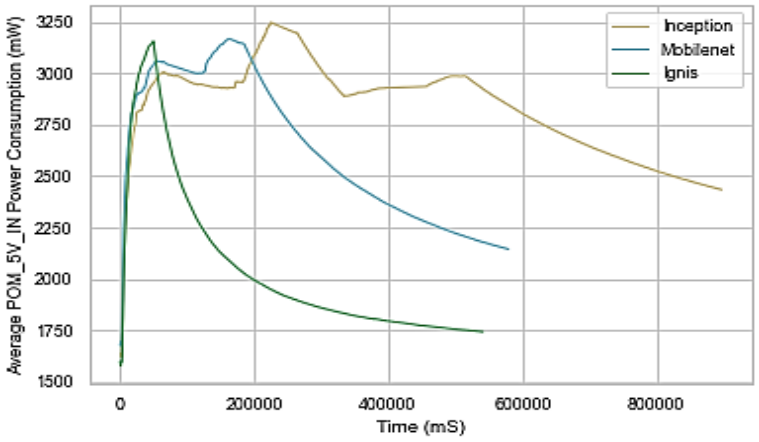
9.2.2 Eficiencia

El proceso de evaluación fue diseñado cuidadosamente para garantizar la precisión de los resultados. Cada algoritmo fue evaluado en un día diferente para evitar la interferencia de factores externos como la temperatura y el uso de la memoria RAM. En el caso de las CNN, el primer paso fue cargar la librería Tensorflow (Ahmed, 2017) para ejecutar el algoritmo de identificación. Sin embargo, este paso no fue necesario para el algoritmo IGNIS, ya que no requiere dicha librería.

Una vez que la tarjeta está en estado de reposo, se lleva a cabo el proceso de medición, que registra los datos de consumo mientras el algoritmo de identificación está en ejecución. No se ejecutan procesos adicionales aparte de los básicos que la tarjeta posee en su inicio regular. Los resultados de la medición se pueden observar en las ilustraciones 22, 23, 24 y 25 correspondientes para cada métrica.

Figura 22.

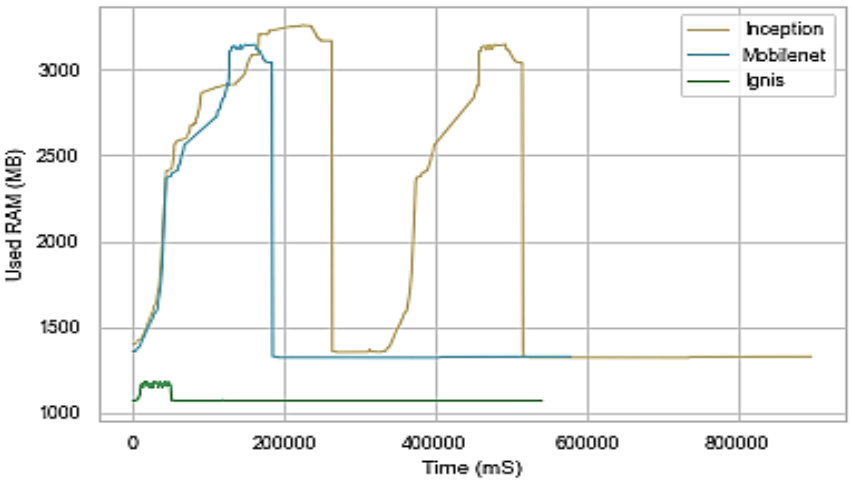
Consumo Energético



La ilustración 22 muestra que el consumo de energía de los tres algoritmos en su ejecución en la tarjeta embebida no supera los 3.25 vatios. Se puede ver que los algoritmos comienzan a disminuir su consumo de la siguiente manera: Inceptionv3 alrededor de los 500 segundos, MobilenetV2 alrededor de los 190 segundos, e IGNIS alrededor de los 50 segundos. Esta disminución indica que la ejecución del algoritmo ha concluido, lo que libera recursos computacionales en la tarjeta y reduce gradualmente el consumo de energía.

Figura 23.

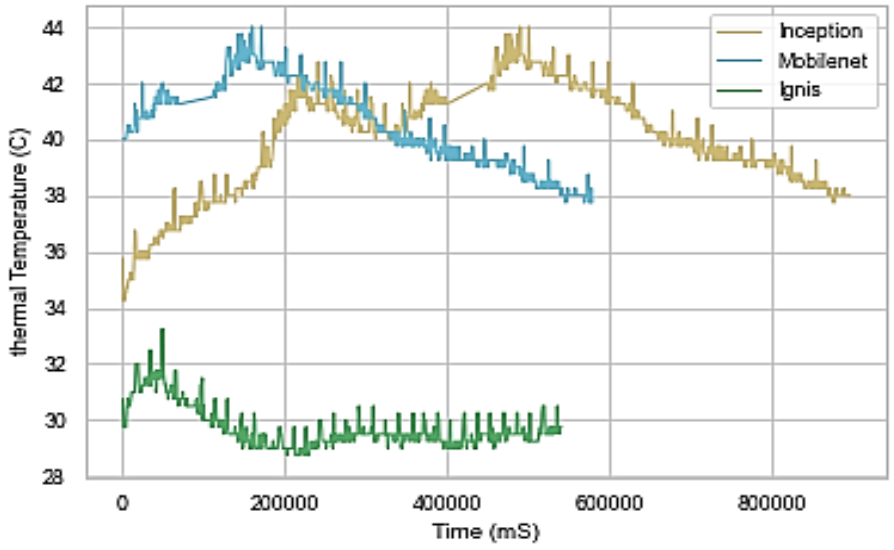
Consumo RAM.



En la Ilustración 23, se puede ver que los algoritmos CNN experimentan picos en su uso de memoria, alcanzando los 3000 MB. Esto comienza a partir de los 1500 MB, probablemente debido a la carga de la librería Tensorflow. Por otro lado, IGNIS presenta un menor uso de memoria, alcanzando aproximadamente los 1200 MB. Esto se debe a que IGNIS no realiza convoluciones, lo que requiere un uso mayor de memoria. Además, se puede ver que los tres algoritmos experimentan un descenso en su uso de memoria aproximadamente en los mismos momentos que se mencionaron en la Ilustración 22, lo que sugiere una relación entre el consumo de energía y la memoria utilizada.

Figura 24.

Temperatura.



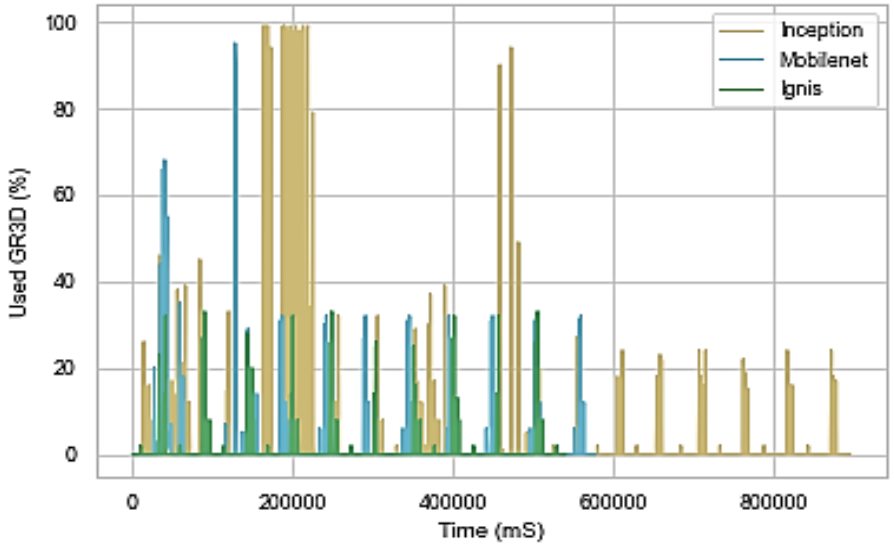
La ilustración 24 grafica la evolución de la temperatura de la tarjeta embebida durante la ejecución de los algoritmos de identificación. Se observa que el algoritmo InceptionV3 presenta un aumento gradual de la temperatura desde los 34°C hasta los 44°C en los primeros 500 segundos, seguido de un descenso paulatino. Por otro lado, el algoritmo MobilenetV2 muestra un aumento similar desde los 40°C hasta los 44°C en los primeros 190 segundos. En contraste, el algoritmo

IGNIS mantiene una estabilidad entre los 29°C y los 32°C, sin presentar elevaciones bruscas como las observadas en los otros dos algoritmos.

La ejecución de los tres algoritmos de identificación en la tarjeta embebida produce un impacto diferente en la temperatura de la tarjeta. Mientras que InceptionV3 y MobilenetV2 presentan una pendiente ascendente en la temperatura, IGNIS mantiene una estabilidad sin elevaciones abruptas. Esto sugiere que IGNIS es más eficiente en términos de gestión de recursos y puede ser una opción más adecuada para aplicaciones que requieran un funcionamiento estable de la temperatura.

Figura 25.

Consumo GPU.



En la ilustración 25 se puede apreciar que, durante la ejecución de los algoritmos en la tarjeta embebida, el uso de la GPU varió. En el caso de InceptionV3, se encontró que el uso estuvo entre el 20% y el 40%, con picos en el 100%. Por otro lado, el uso de la GPU para el algoritmo MobilenetV2 osciló entre el 20% y el 40%, con puntos de mayor utilización que superaron el 80%.

Finalmente, para el algoritmo IGNIS se encontró que la GPU fue usada entre 0% y 30% durante todo el proceso.

El uso de la GPU de la tarjeta embebida varía en el caso de los tres algoritmos evaluados. Mientras que los algoritmos InceptionV3 y MobilenetV2 tuvieron un uso de la GPU muy variado durante el tiempo de ejecución, promediándose por debajo del 50%, pero alcanzando hasta el 100% en algunos momentos, el algoritmo IGNIS tuvo un uso de la GPU menor a 35% todo el tiempo, lo que indican que la tarjeta embebida tiene suficiente capacidad para ejecutar otros programas simultáneamente, debido a que el consumo de memoria GPU y RAM durante la ejecución de IGNIS no supera un nivel crítico.

En conclusión, los resultados obtenidos a partir de las mediciones de eficiencia muestran que el consumo energético de los tres algoritmos en su ejecución sobre la tarjeta embebida no supera los 3,25 watts, y que los algoritmos inician su descenso de consumo aproximadamente a los 500 segundos (Inceptionv3), 190 segundos (MobilenetV2), y 50 segundos (IGNIS). Los algoritmos convolucionales presentan crestas sobre los 3000 MB de RAM, e IGNIS, que no realiza convoluciones, inicia su ascenso desde los 1100 MB y llega a los 1200 MB aproximadamente. La temperatura de la tarjeta embebida presenta una pendiente ascendente durante la ejecución del algoritmo InceptionV3 y MobilenetV2 sin superar los 44°C, mientras que el algoritmo IGNIS se mantiene estable por debajo de los 33°C. Finalmente, el uso de la GPU de la tarjeta embebida oscila entre el 20% y el 40% en el caso de InceptionV3 y MobilenetV2 con elevaciones abruptas, y entre 0% y 30% en el caso de IGNIS. En general, estos resultados son importantes ya que muestran la posibilidad de ejecutar los tres algoritmos en la tarjeta embebida sin un consumo excesivo de memoria RAM, GPU y energía, y manteniendo una temperatura estable.

Para evaluar la diferencia entre los algoritmos, se utiliza como referencia el rendimiento de la tarjeta Jetson Nano y se establece como valor máximo 100%. Se calcula el promedio de cada métrica de cada algoritmo, basado en los datos de medición (ver tabla 4).

Tabla 4.

Promedios Mediciones

Ítems	Jetson Nano	InceptionV3	MobilenetV2	IGNIS
Energía (mW)	5.000	2.825	2.601	2.049
RAM (MB)	4.000	1.957	1.637	1.077
Temperatura (°C)	80	40	40	30
GPU (%)	100	40	23	11

La tabla 4 muestra una comparación de los datos medidos de energía, RAM, temperatura y uso de GPU en la tarjeta Jetson Nano y los algoritmos InceptionV3, MobilenetV2 e IGNIS. Según los datos, se puede concluir que el algoritmo IGNIS tiene el menor consumo de energía, RAM y también la menor temperatura durante su ejecución, mientras que su uso de GPU es el más bajo. Por otro lado, el algoritmo InceptionV3 tiene un consumo de energía y uso de GPU mayor en comparación con los otros dos algoritmos, aunque su temperatura y uso de RAM se encuentran en un rango similar. En general, estos datos muestran cómo cada algoritmo afecta de manera diferente al uso de los recursos de la tarjeta Jetson Nano.

En la Tabla 5 se presentan los resultados promedios de las métricas de eficiencia para la tarjeta Jetson Nano durante la ejecución de cada uno de los algoritmos, expresados en términos de porcentajes de consumo. Estos datos permiten tener una visión generalizada del comportamiento de la tarjeta en relación a su eficiencia durante la ejecución de los algoritmos.

Tabla 5.

Resultados Eficiencia

Algoritmos	Consumo Energía	Consumo RAM	Lectura Temperatura	Consumo GPU	Avg
InceptionV3	57%	49%	50%	40%	49%
MobilenetV2	52%	41%	50%	23%	42%
IGNIS	41%	27%	38%	11%	29%

Se puede concluir que los tres algoritmos tienen un consumo moderado de los recursos de la tarjeta Jetson Nano, con un promedio que no supera el 50%. Sin embargo, en comparación, el algoritmo MobilenetV2 tiene una eficiencia ligeramente mejor en términos de consumo de energía y RAM en comparación con InceptionV3, pero IGNIS presenta un ahorro aún mayor en los mismos aspectos. El consumo de GPU es el recurso con una carga más baja entre todos y el consumo de energía el más altos entre todas las métricas.

Al evaluar los tres algoritmos se observa que el algoritmo InceptionV3 presenta el tiempo de ejecución más prolongado, con un tiempo aproximado de 500 segundos. Por otro lado, el algoritmo IGNIS es el que presenta un tiempo de ejecución más corto, con un tiempo aproximado de 50 segundos. Con respecto a la confiabilidad, el algoritmo IGNIS es el que presenta un promedio más alto, con un 98% de confiabilidad. Finalmente, la eficiencia se refiere a la relación entre el consumo de recursos y la capacidad de procesamiento y en este aspecto, el algoritmo InceptionV3 es el que presenta un promedio más alto, con un 49% de consumo total, pero IGNIS presenta el más bajo con un consumo del 29%.

Se puede apreciar en la evaluación que los resultados obtenidos con las redes convolucionales se ajustan a los niveles de confiabilidad de los resultados presentados en otras investigaciones de clasificación (Phung, & Rhee., 2019). Además, el algoritmo IGNIS demuestra

una confiabilidad acorde con las expectativas de un modelo de identificación, comparándolo favorablemente con los modelos que emplean CNN (Zheng, y otros, 2019).

Este proceso de evaluación se enfocó en valorar los tres algoritmos propuestos para ser parte del modelo de identificación de incendios. El objetivo era seleccionar el algoritmo que arrojara los mejores resultados en cuanto a confiabilidad y eficiencia. Aunque el proyecto no tenía como objetivo principal comparar diferentes algoritmos de clasificación, era necesario evaluar cuál de los tres algoritmos se comportaba mejor para incluirlo en el modelo final propuesto.

10. Conclusiones

Este estudio aporta una novedad al ser el primero en utilizar imágenes satelitales con bandas espectrales M3-I3-M11 para la identificación de incendios. Estas imágenes han demostrado ser efectivas debido al contraste claro entre la conflagración y el fondo de la imagen. Esto abre nuevas posibilidades para el uso de estas imágenes en futuros estudios en el campo de la identificación de incendios.

El algoritmo IGNIS se ha seleccionado como parte del modelo propuesto para la identificación de incendios debido a su rendimiento sobresaliente en la evaluación realizada. Con una confiabilidad del 98%, el algoritmo logró un consumo computacional y energético del 29% en la tarjeta embebida Nvidia Jetson Nano, demostrando su eficiencia en términos de recursos y su capacidad para ser implementado en aplicaciones de detección de incendios en tiempo real, esto sumado a que no necesita entrenamiento previo, como lo requieren las CNN.

Se estableció un flujo de trabajo con etapas claras y específicas que facilitan la reproducción o adaptación del modelo propuesto para las necesidades de diversos usuarios.

En esta investigación, se demostró la viabilidad de utilizar la tarjeta Jetson Nano como plataforma para la ejecución de modelos de visión por computadora y/o aprendizaje profundo. Esta tarjeta ofrece una combinación atractiva de potencia de cálculo y bajo costo, lo que la hace una opción valiosa para los desarrolladores. Además, cuenta con una herramienta para medir el consumo de recursos, lo que facilita el monitoreo de los modelos en términos de eficiencia.

La evaluación de las redes neuronales en este proyecto, basadas en imágenes satelitales de mayor tamaño en comparación con las imágenes utilizadas en su entrenamiento, sugiere una posible relación entre la confiabilidad de las predicciones y la discrepancia de tamaño entre las imágenes de entrenamiento e inferencia. Es importante realizar investigaciones adicionales para verificar esta hipótesis y obtener una comprensión más profunda de este tema, sin embargo, esta investigación no abarca ese aspecto.

11. Trabajos Futuros

Tras los resultados de este proyecto, se proponen tres líneas de trabajo futuro. La primera consiste en personalizar la metodología para desarrollar múltiples modelos de clasificación de incendios con un enfoque en la eficiencia computacional y la precisión de las predicciones, a un bajo costo. La segunda consiste en ampliar los conjuntos de datos abiertos disponibles para ayudar a los investigadores y agencias a prevenir incendios. Finalmente, este proyecto está en desarrollo e implementación para su uso real dentro de la Misión Espacial como Modelo de Servicios (SMMAS), un modelo de cómputo en la nube HPC para proyectos espaciales desarrollado por el Centro de Computación Científica y de Alto Rendimiento de la Universidad Industrial de

Santander (SC3UIS). Algunos de las actividades a ejecutar para continuar con esta línea de investigación son:

- Establecer un proceso integral para la conversión de datos recibidos de satélites, en imágenes con bandas espectrales M3-I3-M11 que se integren directamente en los desarrollos individuales, a fin de optimizar la eficiencia y el flujo de trabajo en la investigación.
- Llevar a cabo evaluaciones de IGNIS en otras áreas de investigación, como la agricultura de precisión, la medicina, el desarrollo urbano, y en general, en cualquier situación en la que el color de un objeto sea un factor clave en su identificación y diferenciación.
- Implementar el algoritmo de IGNIS en diferentes plataformas de hardware embebido con el objetivo de evaluar el desempeño y los requisitos mínimos de computación necesarios para su funcionamiento. Además, se pueden realizar pruebas para evaluar el tiempo de respuesta en diferentes niveles de recursos computacionales.

Referencias

Ahmed, G. (2017). *TensorFlow: A Guide To Build Artificial Neural Networks Using Python.*

Alonso, A., & Millán, F. (2019). INVIERNO: Anticiclón peninsular. *Revista Tiempo y Clima, 5(68)*.

Arguello, J., Hernández, C., & Ferreira, J. (2022). *Towards Fire Identification Model in Satellite Images Using HPC Embedded Systems and AI.* . Springer, Cham. : In Latin American High Performance Computing Conference (pp. 103-115). .

Ba, R., Chen, C., Yuan, J., Song, W., & Lo, S. (2019). *SmokeNet: Satellite smoke scene detection using convolutional neural network with spatial and channel-wise attention.* Remote Sensing, 11(14), 1702.

Benítez, R. (2014). *Inteligencia artificial avanzada.* . Obtenido de UOC. : https://bibliotecavirtual.uis.edu.co:4259/es/lc/uis/titulos/57582

Beysolow II, T. (2017). *Introduction to Deep Learning Using R.* . Obtenido de https://bibliotecavirtual.uis.edu.co:2142/book/10.1007/978-1-4842-2734-3

Briess, K., Jahn, H., & et.al. (2003). *Fire recognition potential of the Bi-spectral InfraRed Detection (BIRD) satellite.* . Int. J. Remote Sensing 24(4), 865-872.

Brozek, T. (2014). *Micro-and nanoelectronics: emerging device challenges and solutions.* . CRC Press.

Bruijning, M., Visser, M. D., Hallmann, C. A., Jongejans, E., & Golding, N. (2018). trackdem: Automated particle tracking to obtain population counts and size distributions from videos in r. *Methods in Ecology and Evolution.*, 965–973. doi:10.1111/2041-210X.12975

Caicedo Bravo, E., & López Sotelo, J. (2009). *Una aproximación práctica a las redes neuronales artificiales.* . Obtenido de Programa Universidad del Valle. : https://bibliotecavirtual.uis.edu.co:4259/es/lc/uis/titulos/129183

Cao, C., Blonski, S., Wang, W., Uprety, S., Shao, X., Choi, J., & ... Kalluri, S. (2018). *November). NOAA-20 VIIRS on-orbit performance, data quality, and operational Cal/Val support. In Earth Observing Missions and Sensors: Development, Implementation, and Characterization V (Vol. 10781, p. 107810K).* International Society for Optics and Photonics.

Cao, C., Xiong, J., Blonski, S., Liu, Q., Uprety, S., Shao, X., & Weng, F. (2013). Suomi NPP VIIRS sensor data record verification, validation, and long-term performance monitoring. *ournal of Geophysical Research: Atmospheres, 118*(20), 664-678. doi:https://doi.org/10.1002/2013JD020418

Casas Roma, J., Bosch Rué, A., & Lozano Bagén, T. (2019). *Deep learning: principios y fundamentos.* . Obtenido de UOC: https://bibliotecavirtual.uis.edu.co:4259/es/lc/uis/titulos/126167

De Silva, D., & Damminda, A. (2022). *An artificial intelligence life cycle: From conception to production.* . Patterns. 100489.

Deng, J., Dong, W., Socher, R., Li, L., Li, K., & Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 248-255. doi:10.1109/CVPR.2009.5206848

Digital Globe. (sf). Obtenido de https://discover.digitalglobe.com/

Feng, X., Jiang, Y., Yang, X., Du, M., & Li, X. (2019). Computer vision algorithms and hardware implementations. *A survey. Integration, 69*, 309-320.

Field Programmable Gate Array (FPGA). (sf). Obtenido de https://www.arm.com/glossary/fpg

Ganapathi Subramanian, S., & Crowley, M. (2018). *Using spatial reinforcement learning to build forest wildfire dynamics models from satellite images.* . Frontiers in ICT, 5, 6.

Github. (sf). *Proyecto-IGNIS*. Obtenido de https://github.com/jhonesis/Proyecto-IGNIS

Graphic Processing Unit (GPU). (sf). Obtenido de https://www.intel.la/content/www/xl/es/products/docs/processors/what-is-a-gpu.html

Halle, W., Fischer, C., Terzibaschian, T., Zell, A., & Reulke, R. (2019). *Infrared-Image Processing for the DLR FireBIRD Mission.* Springer, Singapore: In Asian Conference on Pattern Recognition (pp. 235-252). .

Hatekar, A., Manwani, S., Patil, G., & Parekh, A. (2017). *Fire detection on a surveillance system using Image processing.* . International Journal of Engineering Research & Technology (ijERT), 6(05).

Hernandez, E., Montenegro, C., & Navaux, P. (2016). EnerGyPU and enerGyPhi monitor for power consumption and performance evaluation on nvidia tesla GPU and Intel Xeon Phi. . *In 2016 16th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid). IEEE,* 718-725.

Huang, T. S. (s.f.). Computer Vision: Evolution and Promise. *19th CERN School of Computing*, 21–25. doi:10.5170/CERN-1996-008.21

Huang, T. (sf). *Computer Vision: Evolution and Promise,» 19th CERN School of Computing.*

Jin, C., Tanno, R., Mertzanidou, T., Panagiotaki, E., & Alexander, D. (2021). *Learning to downsample for segmentation of ultra-high resolution images. arXiv preprint arXiv:2109.11071.*

Kadochnikov, A. (2019). *Software and Technological Support of Geoinformation Web System for the Operative Processing and Visualization of Satellite Data.* . In CEUR Workshop Proceedings (pp. 309-315).

Khryashchev, V., & Larionov, R. (2020). *Wildfire segmentation on satellite images using deep learning.* . In 2020 Moscow Workshop on Electronic and Networking Technologies (MWENT) (pp. 1-5). IEEE.

Knight, E., & Kvaran, G. (2014). *Landsat-8 operational land imager design, characterization and performance.* . Remote sensing, 6(11), 10286-10305.

Langford, Z., Kumar, J., & Hoffman, F. (2018). *Wildfire mapping in Interior Alaska using deep neural networks on imbalanced datasets.* . In 2018 IEEE International Conference on Data Mining Workshops (ICDMW) (pp. 770-778). IEEE.

Lorenz, E. (2013). *Thermal Remote Sensing with Small Satellites: BIRD, TET and Next Generation BIROS In: Thermal Infrared Remote Sensing.* . New York London: Springer Dordrecht Heidelberg New York London, ed. C. Kuenzer & St. Dech.

Maggiori, E., Tarabalka, Y., Charpiat, G., & Alliez, P. (2017). *Can semantic labeling methods generalize to any city? the inria aerial image labeling benchmark.* . In 2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS) (pp. 3226-32.

Maity, A. (2015). Improvised Salient Object Detection and Manipulation. *I.J. Image, Graphics and Signal Processing*. Obtenido de https://arxiv.org/ftp/arxiv/papers/1511/1511.02999.pdf

Maity, A. (2015). *Improvised Salient Object Detection and Manipulation.* I.J. Image, Graphics and Signal Processing, .

Manaswi, N. (2018). *Deep Learning with Applications Using Python.* . Obtenido de https://bibliotecavirtual.uis.edu.co:2142/book/10.1007/978-1-4842-3516-4

Meseguer González, P., López, R., & Badia, d. M. (2017). *Inteligencia artificial.* . Obtenido de Editorial CSIC Consejo Superior de Investigaciones Científicas: https://bibliotecavirtual.uis.edu.co:4259/es/lc/uis/titulos/42319

Moderate Resolution Imaging Spectroradiometre (MODIS). (sf). Obtenido de https://modis.gsfc.nasa.gov/

National Aeronautics and Space Administration. (2020). *data visualization application WORLDVIEW*. Obtenido de https://worldview.earthdata.nasa.gov/

National Aeronautics and Space Administration. (s.f.). *data visualization application WORLDVIEW*. Recuperado el 20 de octubre de 2020, de https://worldview.earthdata.nasa.gov/

National Oceanic and Atmospheric Administration. (2017). *NOAA Technical Report NESDIS 142*. Obtenido de https://ncc.nesdis.noaa.gov/documents/documentation/viirs-users-guide-tech-report-142a-v1.3.pdf

Nvidia. (s.f.). *Jetson Nano*. Obtenido de https://nvidia.com/en-us/autonomus-machines/embedded-systems/jetson-nano/

Open Computer Vision,. (sf). Obtenido de https://opencv.org/

Petrescu, R. V., & et.al. (2018). NASA satellites help us to quickly detect forest fires. *Ameri-can Journal of Engineering and Applied Sciences, vol. 11, no 1,* 288-296.

Phung, & Rhee. (2019). *A High-Accuracy Model Average Ensemble of Convolutional Neural Networks for Classification of Cloud Image Patches on Small Datasets. Applied Sciences, 9(21), 4500. MDPI AG*. Obtenido de http://dx.doi.org/10.3390/app9214500

Sahoo, S. (2021). *How to Train MobileNetV2 On a Custom Dataset*. Recuperado el 24 de Junio de 2021, de https://blog.roboflow.com/how-to-train-mobilenetv2-on-a-custom-dataset/

Sandler, M., Howard, A. G., Zhu, M., Zhmoginov, A., & Chen, L. (2018). MobileNetV2: Inverted Residuals and Linear Bottlenecks. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4510-4520.

Sarkar, D., Bali, R., & Ghosh, T. (2018). *Hands-On Transfer Learning with Python: Implement advanced deep learning and neural network models using TensorFlow and Keras.* Packt Publishing.

Schmidt, C., Prins, E., & Feltz, J. (2002). *The GOES wildfire automated biomass burning algorithm processing system.* In AGU Spring Meeting Abstracts. pp. A21B-07).

Schroeder, W., Oliva, P., Giglio, L., Quayle, B., Lorenz, E., & Morelli, F. (2016). *Active fire detection using Landsat-8/OLI data, Remote Sensing of Environment, Volume 185, Pages 210-220, ISSN 0034-4257*. Obtenido de https://doi.org/10.1016/j.rse.2015.08.032

Shah, D. (2020). *Early Fire detection system using deep learning and OpenCV*. Recuperado el 20 de Junio de 2021, de https://towardsdatascience.com/early-fire-detection-system-using-deep-learning-and-opencv-6cb60260d54a

Sims, E. (2021). *Assessment of GOES-16 Satellite and High-Resolution Rapid Refresh Model Data for Blowing Snow Impact-Based Decision Support Services* . (Doctoral dissertation, The University of North Dakota).

Soltani, A. A., Huang, H., Wu, J., Kulkarni, T. D., & Tenenbaum, J. B. (2017). Synthesizing 3D Shapes via Modeling Multi-View Depth Maps and Silhouettes With Deep Generative Networks. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1511–1519. doi:10.1109/CVPR.2017.269

Soltani, A., Huang, H., Wu, J., Kulkarni, T., & Tenenbaum, J. (2017). *Synthesizing 3D Shapes via Modeling Multi-View Depth Maps and Silhouettes With Deep Generative Networks.* Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition .

Space Net. (sf). Obtenido de https://spacenet.ai/

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the Inception Architecture for Computer Vision. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2818-2826.

Talebi, H., & Milanfar, P. (2021). *Learning to resize images for computer vision tasks.* . In Proceedings of the IEEE/CVF international conference on computer vision (pp. 497-506).

Trambitckii, K., Anding, K., Musalimov, V., & Linss, G. (2015). Colour based fire detection method with temporal intensity variation filtration. *In Journal of Physics: Conference Series (Vol. 588, No. 1). IOP Publishing*, 012038.

Wang, X., Ng, H., & Liang, J. (2017). *Lapped convolutional neural networks for embedded systems.* In 2017 IEEE Global Conference on Signal and Information Processing (GlobalSIP) (pp. 1135-1139). IEEE.

Weaver, J., Lindsey, D., Bikos, D., Schmidt, C., & Prins, E. (2004). *Fire Detection Using GOES Rapid Scan Imagery, Weather and Forecasting, 19(3).*

Weng, F. (2018). *7.03 - Advanced Technology Microwave Sounder Calibration and Validation, Editor(s): Shunlin Liang, Comprehensive Remote Sensing, Elsevier, Pages 42-63, ISBN 9780128032213,* . Obtenido de https://doi.org/10.1016/B978-0-12-409548-9.10393-8

Wiki. (s.f.). *SuperComputación y Cálculo Científico*. Obtenido de http://wiki.sc3.uis.edu.co/index.php/Wiki_SC3

Zheng, W., Zhang, X., Kim, J. J., Zhu, X., Ye, G., Ye, B., . . . Si, J. M. (2019). *High Accuracy of Convolutional Neural Network for Evaluation of Helicobacter pylori Infection Based on Endoscopic Images: Preliminary Experience, Clinical and Translational Gastroenterology: Volume 10 - Issue 12* -. Obtenido de 10.14309/ctg.0000000000000109

Zoph, B., Cubuk, E., Ghiasi, G., Lin, T., Shlens, J., & Le, Q. (2019). Learning data augmentation strategies for object detection. arXiv. Obtenido de arXiv:1906.11172