ISSN: **2773-7330**
Vol. 4 No. 3.1 pp. 109 – 126. agosto 2022

www.alfapublicaciones.com

Eficiencia comparativa en la ejecución de algoritmos de visión artificial

Comparative efficiency in the execution of artificial vision algorithms

[1] Oswaldo Martínez Guashima https://orcid.org/0000-0001-9018-7777
Escuela Superior Politécnica de Chimborazo (ESPOCH), Facultad de Informática y Electrónica. Riobamba-Ecuador
o.martinez@espoch.edu.ec

[2] Jorge Paucar Samaniego https://orcid.org/0000-0002-1704-8583
Escuela Superior Politécnica de Chimborazo (ESPOCH), Facultad de Informática y Electrónica. Riobamba-Ecuador
jlpaucar@espoch.edu.ec

[3] Christian Alberto Costales Espinoza https://orcid.org/0000-0001-8021-1489
Universidad Estatal de Bolívar (UEB). Guaranda-Ecuador
ccostales@ueb.edu.ec

[4] Christian Fernando Barragán Quizhpe https://orcid.org/0000-0003-4699-9553
Universidad Estatal de Bolívar (UEB). Guaranda-Ecuador
cbarragan@ueb.edu.ec

Artículo de Investigación Científica y Tecnológica
Enviado: 12/05/2022
Revisado: 27/06/2022
Aceptado: 15/07/2022
Publicado:05/08/2022
DOI: **https://doi.org/10.33262/ap.v4i3.1.241**

Cítese: Martínez Guashima, O., Paucar Samaniego, J., Costales Espinoza, C. A., & Barragán Quizhpe, C. F. (2022). Eficiencia comparativa en la ejecución de algoritmos de visión artificial. AlfaPublicaciones, 4(3.1), 109–126. https://doi.org/10.33262/ap.v4i3.1.241

ALFA PUBLICACIONES, *es* una Revista Multidisciplinar, **Trimestral,** que se publicará en soporte electrónico tiene como **misión** contribuir a la formación de profesionales competentes con visión humanística y crítica que sean capaces de exponer sus resultados investigativos y científicos en la misma medida que se promueva mediante su intervención cambios positivos en la sociedad. https://alfapublicaciones.com
La revista es editada por la Editorial Ciencia Digital (Editorial de prestigio registrada en la Cámara Ecuatoriana de Libro con No de Afiliación 663) www.celibro.org.ec

ISSN: **2773-7330**
Vol. 4 No. 3.1 pp. 109 – 126. agosto 2022

www.alfapublicaciones.com

Palabras claves:
aceleradores de redes neuronales, raspberry pi 4, redes neuronales, visión artificial.

Resumen

Introducción: Una red neuronal es un algoritmo programado con base en un modelo de inferencia Para la ejecución de un proceso de visión es necesario una inferencia avanzada también es deseable un bajo consumo de energía **Objetivos:** analizar la eficiencia comparativa en la ejecución de visión artificial mediante sistemas embebidos. **Metodología:** Comparar los tiempos de inferencia que conlleva el ejecutar una red neuronal profunda en un sistema embebido del tipo Raspberry Pi 4 y un acelerador de inferencias diseñado por Intel **Resultados:** Se detalla el proceso de instalación y configuración necesaria para la compatibilidad entre los dos dispositivos, se utiliza modelos de redes neuronales ya entrenadas enfocadas al procesamiento de imágenes y se compara el tiempo de inferencia que conlleva el ejecutarlos **Conclusiones:** Al final se llega a la conclusión que para estas condiciones experimentales el tiempo de inferencia se ha mejorado en un 25 %.

Keywords:
neural network accelerators, raspberry pi 4, neural networks, artificial vision.

Abstract

Introduction: A neural network is a programmed algorithm based on an inference model. For the execution of a vision process, advanced inference is necessary. Low energy consumption is also desirable. **Objectives**: to analyze the comparative efficiency in the execution of artificial vision through embedded systems. **Methodology**: To compare the inference times involved in executing a deep neural network in an embedded system such as Raspberry Pi 4 and an inference accelerator designed by Intel. **Results:** The installation and configuration process necessary for compatibility between the two devices is detailed already trained neural network models focused on image processing are used and the inference time involved in executing them is compared. **Conclusions**: In the end, it is concluded that for these experimental conditions the inference time has been improved by 25%.

Introducción

Una red neuronal es un algoritmo programado con base en un modelo de inferencia cuyo objetivo es imitar el funcionamiento de un sistema neuronal

humano y que puede interactuar con otras similares intercambiando información con el fin de cumplir un objetivo y representarlo en salidas continuas o discretas.

Una de las aplicaciones más comunes es aquellas en las cuales interviene el procesamiento de imágenes ejecutadas en sistemas micro procesados. La visión por computador permite adquirir una imagen usando dispositivos electrónicos para procesarla y obtener un resultado en específico como por ejemplo una clasificación. Todo este proceso conlleva un con- junto de etapas que requieren un procesamiento computacional elevado.

Para la ejecución de un proceso de visión es necesario una inferencia avanzada también es deseable un bajo consumo de energía en el hardware y un rendimiento elevado, sin embargo, la mayoría de los dispositivos poseen un nivel de procesamiento limitado. Para redes neuronales simples estas limitaciones no influyen en sus resultados, pero, para sistemas complejos de inteligencia artificial, se necesita dispositivos de coprocesamiento. En los últimos años muchas empresas han desarrollado dispositivos dedicados a procesar programas de inteligencia artificial, empresas como Google, Intel o Nvidia poseen en su portafolio de productos dispositivos que reducen los altos requerimientos de procesamiento.

Estado del arte

En esta sección se analiza la información que permita identificar los requerimientos de un sistema de inteligencia artificial enfocado a procesos de visión, para lo cual se estudia los conceptos y teorías útiles para el desarrollo.

Redes Neuronales

Una red neuronal asimila información de forma jerarquizada y por fases, esta se define en función del número de capas que contenga la red siendo las primeras aquellas que aprenden conceptos muy concretos y las capas subsiguientes las que aprenden conceptos abstractos. El proceso de aprendizaje de una red neuronal se establece en dos fases que comprenden (Vázquez, 2020):

1) *Fase de entrenamiento*: en la cual ingresa la información etiquetada con el fin de construir el modelo, a su vez necesita una cantidad de datos considerable que permitan establecer características de entrada y salida.
2) *Fase de inferencia*: una vez que el modelo ha sido desarrollado y se han establecido etiquetas se debe realizar predicciones de los datos no etiquetados.

Arquitectura de una red neuronal

La arquitectura básica, con una entrada y una salida, de una red neuronal se puede

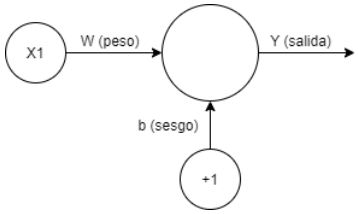
ISSN: **2773-7330**
Vol. 4 No. 3.1 pp. 109 – 126. agosto 2022

www.alfapublicaciones.com

divisar en la figura 1, en esta se encuentra la ubicación de la entrada X1, la salida y, el bías b y su peso w. (Rengifo, 2002).

Figura 1

Arquitectura neurona artificial



Función de activación: Como menciona Garduño & Padrón (2000), una red neuronal artificial se puede aproximar de manera matemática a una función lineal o a una no lineal. Para las aproximaciones no lineales se utiliza funciones de activación que permiten disminuir la complejidad matemática en el sistema, lo que implica que la salida de la red está es la función de la ecuación 1.

$$y = \text{funcion\_activacion}(x) \qquad (1)$$

Taza de aprendizaje: Es necesario considerar un parámetro que influye en la modificación de los pesos w y el bías b del modelo, se lo conoce como learning rate su símbolo es η y determina la velocidad con la que se modifica sus valores y el error (Jiménez, 2012). Este parámetro es esencial en redes que se entrenan con backpropagation el cual se considera una herramienta para disminuir el error generado ya que la relación entre estos parámetros es directamente proporcional como se define en la ecuación 2.

$$\Delta\omega = \omega\,(t+1) - \omega(t) = \eta \cdot \text{error} \qquad (2)$$

Algoritmo backpropagation: En el entrenamiento de una red neuronal se pueden utilizar varios algoritmos que, dependiendo de la aplicación, pueden modificar su eficiencia (Matich, 2001). Entre estos algoritmos se encuentra el denominado como backpropagation que consiste en una propagación de errores hacia atrás, usando el valor del error generado en la salida de una red neuronal artificial multicapa para ajustar los parámetros que involucran la respuesta a una entrada, esto genera que se reduzca el error en cada interacción (Juárez & Pascual, 2015).

Redes neuronal convolucionales

Una CNN es una arquitectura de uso común en procesos de aprendizaje profundo, se usa en sistemas de visión artificial cuya característica principal de su configuración multicapa permiten procesar imágenes de entrada y obtener, con mayor detalle, sus

ISSN: **2773-7330**
Vol. 4 No. 3.1 pp. 109 – 126. agosto 2022

www.alfapublicaciones.com

características (Ordoñez, 2020).

El proceso de convolución consiste en procesar imágenes aplicando filtros, conocidos también como kernels, específicos con el fin de extraer características, varias máscaras producirán resultados diferentes debido a una combinación lineal de los pixeles de entrada (Espinola et al., 2011).

Hardware para IA

En Gómez & Vélez (2015), se detalla que en aplicaciones de inteligencia artificial es necesario encontrar el hardware correcto que permita la ejecución de los algoritmos de manera eficiente. Es importante considerar parámetros como la capacidad de almacenamiento de los sistemas microprocesados, su velocidad de procesamiento y su capacidad de realizar procesamiento en paralelo tomando en cuenta que se ejecutará inferencias complejas por lo cual no se debe sacrificar precisión en sus resultados.

Aceleradores: En el artículo presentado por Montoya (2020), se detalla que un acelerador de inteligencia artificial es un dispositivo embebido dedicado a ejecutar aplicaciones de redes neuronales, visión artificial y aprendizaje automático. Las aplicaciones consisten en la ejecución de algoritmos complejos usando una gran cantidad de datos o el barrido de dispositivos externos, como sensores, que requieren una elevada capacidad de cálculo con procesamiento en paralelo. Las CPU tradicionales pueden ser utilizadas para este propósito, sin embargo, para la ejecución de algoritmos de inteligencia artificial se requiere cálculos concretos durante la fase de aprendizaje e inferencia con operaciones de matrices muy elaboradas. Una CPU convencional realiza este tipo de operaciones en función del número de núcleos que posee con una limitación importante que es la ejecución de 4 cálculos por ciclo de máquina (CM) acortando así el número de operaciones.

Intel Stick: Es un dispositivo USB que posee una Intel Neural Computer Stick 2 (Intel Movidius Myriad X), que es considerado un motor de cálculo neuronal el cual se aplica en inferencias de redes neuronales profundas dedicadas a procesos de visión artificial. Posee 16 núcleos con una arquitectura VLIW (Very Long Instruction Word lo que ayuda en la ejecución de procesos en paralelo (Fernández, 2020).

No necesita conectarse a la internet ya que se ejecuta la inferencia en tiempo real y su consumo de energía es bajo lo que permite operar con sistemas como la Raspberry Pi. Soporta frameworks como TensorFlow, Caffe, Apache MxNet, Pytorch con una conectividad USB 3.0, como se puede observar en la figura 2.

ISSN: **2773-7330**
Vol. 4 No. 3.1 pp. 109 – 126. agosto 2022

www.alfapublicaciones.com

Figura 2

FrameWorks soportados

Fuente: Intel.com (2022)

El fabricante ha desarrollado un kit de herramientas denominado OpenVINO, este permite optimizar aplicaciones de deep learning en CNN, el software posee soporte para los dispositivos del tipo Neural Computer Stick (Rodríguez & Ruz, 2020), también incluye librerías en funciones de visión artificial y filtros pre optimizados lo que disminuye el tiempo de desarrollo de las aplicaciones e incrementa el rendimiento de los procesadores utilizando un optimizador de modelos y un motor de inferencia.: El optimizador de modelos permite una transición entre los entornos de entrenamiento y despliegue de una red neuronal, ajusta los modelos de deep learning y genera un análisis estático creando una representación intermedia del modelo ya entrenado que puede ser inferida en su motor para esto necesita dos archivos fundamentales (Salazar, 2017).

1) Archivo .xml en el cual se describe la topología de la red.
2) Archivo .bin el cual contiene los pesos y los datos binarios de la red.

Una vez que el optimizador de modelos ha logrado su objetivo el motor de inferencia se encarga de obtener resultados en función de los datos de entrada. Su funcionamiento se basa en una arquitectura de plugins lo cual permite su compatibilidad con hardware de intel, es decir el motor de inferencia puede ser ejecutado en CPU, FPGA o MYRIAD de intel.

Hardware embebido compatible: Es necesario conectar el acelerador de hardware a un sistema microprocesado (em- bebido) usando un puerto de comunicaciones del tipo serial (USB) con el fin de acelerar la inferencia. Uno de estos dispositivos es la Raspberry Pi que es una computadora ensamblada en una sola tarjeta o circuito que, dependiendo del modelo, puede tener características de procesamiento específicas. En este caso se utiliza un modelo Pi 4 como el de la figura 3 con particularidades básicas como:

ISSN: **2773-7330**
Vol. 4 No. 3.1 pp. 109 – 126. agosto 2022

www.alfapublicaciones.com

Figura 3

Raspberry Pi 4

Fuente: Raspberry Pi. (2022)

Procesador: Broadcom BCM2711, Quad Core Cortex A72, arquitectura de 64 bits a 1.5 Ghz.

Ram de 4GB LPDD4-3200

Metodología

Para Evaluar el tiempo de inferencia que conlleva el ejecutar un algoritmo de inteligencia artificial sobre dos tipos de hardware diferente, uno de ellos utiliza un coprocesador del tipo Intel Neural Computer Stick 2. es necesario ejecutar los algoritmos en función del procedimiento descrito en la figura 4.

Se opta por utilizar dos IDEs diferentes el primero Jupyter y el segundo la consola de Python de la Raspberry Pi 4.

Modelo por utilizar y target: Es necesario importar las librerías necesarias para manipular la información. Se importan librerías como openCV simbolizado por cv2, se requiere escribir y leer archivos que contienen información para esto se utiliza os, numpy es necesario para manipular vectores y matrices, la librería time permite medir el tiempo de inferencia del algoritmo y por último PIL que permitirá el tratamiento y edición de imágenes.

El modelo se importa desde una ubicación específica con los archivos de terminación .xml y .bin para el motor de inferencia OpenVINO. Se debe especificar que es el único formato permitido en la Intel Neural Computer Stick 2, la selección del target depende del hardware como se puede observar en la figura 5.

Imágenes: Se necesita establecer la dirección o path en la cual se almacenan las imágenes que ingresan a la inferencia del modelo. Es necesario unificar el tamaño de las imágenes con el fin de limitar el número de píxeles de entrada estandarizando así su tamaño.

ISSN: **2773-7330**
Vol. 4 No. 3.1 pp. 109 – 126. agosto 2022

www.alfapublicaciones.com

Figura 4

Procesamiento de inferencia

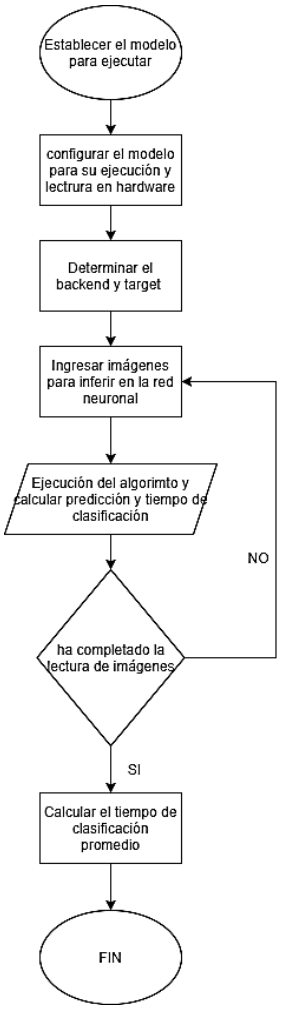
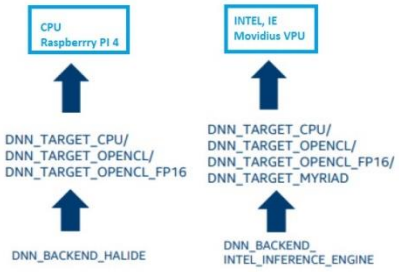


Figura 5

Backends y targets de OpenCV



Ejecución de la inferencia: Con base en el modelo se desarrolla el código de

ISSN: **2773-7330**
Vol. 4 No. 3.1 pp. 109 – 126. agosto 2022

www.alfapublicaciones.com

inferencia en el cual se ingresa la imagen acondicionada antes de la inferencia, se inicia un contador de tiempo que se detendrá luego de obtener el resultado. Para finalizar se genera una variable que permite calcular el tiempo promedio de la clasificación de todas las imágenes que contiene la carpeta utilizando la herramienta de valor medio, mínimo y máximo del conjunto de valores almacenados en el arreglo.

Resultados

Una vez implementados los algoritmos de inferencia sobre los dos tipos de hardware se presenta los resultados obtenidos.

odelo Intel

El modelo utilizado es proporcionado por Intel para la inferencia sobre una Intel Neural Computer Stick 2 y un CPU. La métrica para la detección de personas es del 73.63%, el número de objetos máximos de detección son 200 y el número de operaciones de coma flotante es de 3560 Gflops. El modelo está entrenado en función de 219181 imágenes y para su inferencia necesita una entrada de las características:

name: input.1, shape: [1x3x512x512] - An input image in the format [BxCxHxW].

B batch size

C número de canales.

H altura de la imagen.

W ancho de la imagen.

Raspberry Pi 4: Los resultados de los tiempos de inferencia del algoritmo implementado en el hardware Raspberry Pi 4 se puede observar en la figura 6.

Figura 6

Tiempos de inferencia

```
        if confidence>0.5:
            cv.rectangle(imageAux,(xmin,ymin),(xmax,ymax),color=(0,255,0))
            rostro=imageAux[ymin:ymax,xmin:xmax]
            cv.imwrite('caminar{}.jpg'.format(count),imageAux)
            count=count+1
```

```
El tiempo de ejecución ha sido de 0.34 segundos
El tiempo de ejecución ha sido de 0.428 segundos
El tiempo de ejecución ha sido de 0.299 segundos
El tiempo de ejecución ha sido de 0.299 segundos
El tiempo de ejecución ha sido de 0.291 segundos
El tiempo de ejecución ha sido de 0.35 segundos
El tiempo de ejecución ha sido de 0.289 segundos
El tiempo de ejecución ha sido de 0.457 segundos
El tiempo de ejecución ha sido de 0.462 segundos
El tiempo de ejecución ha sido de 0.366 segundos
El tiempo de ejecución ha sido de 0.31 segundos
El tiempo de ejecución ha sido de 0.313 segundos
El tiempo de ejecución ha sido de 0.344 segundos
El tiempo de ejecución ha sido de 0.46 segundos
El tiempo de ejecución ha sido de 0.38 segundos
El tiempo de ejecución ha sido de 0.305 segundos
```

ISSN: **2773-7330**
Vol. 4 No. 3.1 pp. 109 – 126. agosto 2022

www.alfapublicaciones.com

Las muestras del proceso de inferencia se observan en las figuras 7 que es la imagen de entrada, la figura 8 que es la imagen de salida, cabe recalcar que en esta figura no se ha detectado todas las personas.

Figura 7

Muestra imagen de entrada

Figura 8

Muestra imagen inferida en Raspberry Pi 4

Raspberry Pi 4 e Intel Neural Stick 2: Los resultados de los tiempos de inferencia del algoritmo implementado en el hardware Raspberry pi 4 más Intel Neural Computer Stick 2 intel se puede observar en la figura 9 y la imagen resultante se observa en la figura 10 (Abellán, 2021).

ISSN: **2773-7330**
Vol. 4 No. 3.1 pp. 109 – 126. agosto 2022

www.alfapublicaciones.com

Figura 9

Tiempos de inferencia Raspberry Pi 4 e Intel Neural Stick 2

```
        if confidence>0.5:
            cv.rectangle(imageAux,(xmin,ymin),(xmax,ymax),color=(0,255,0))
            rostro=imageAux[ymin:ymax,xmin:xmax]
            cv.imwrite('caminar{}.jpg'.format(count),imageAux)
            count=count+1
```

```
El tiempo de ejecución ha sido de 0.286 segundos
El tiempo de ejecución ha sido de 0.299 segundos
El tiempo de ejecución ha sido de 0.279 segundos
El tiempo de ejecución ha sido de 0.278 segundos
El tiempo de ejecución ha sido de 0.29 segundos
El tiempo de ejecución ha sido de 0.295 segundos
El tiempo de ejecución ha sido de 0.295 segundos
El tiempo de ejecución ha sido de 0.289 segundos
El tiempo de ejecución ha sido de 0.278 segundos
El tiempo de ejecución ha sido de 0.286 segundos
El tiempo de ejecución ha sido de 0.289 segundos
El tiempo de ejecución ha sido de 0.302 segundos
El tiempo de ejecución ha sido de 0.281 segundos
El tiempo de ejecución ha sido de 0.294 segundos
El tiempo de ejecución ha sido de 0.289 segundos
El tiempo de ejecución ha sido de 0.287 segundos
```

Modelo desarrollado

Esta sección describe el proceso ejecutado con el fin de comparar el tiempo de inferencia entre los dos tipos de hardware ya definidos partiendo de un modelo que se basa en openCV para la detección de rostros en una imagen. Este modelo requiere ser acondicionado con el fin de que se ejecute en la Intel Neural Computer Stick 2. Para facilitar la construcción, entrenamiento e implementación de modelos de topologías comunes como SSD, FAST, CNN o RNN el motor de inferencia OpenVINO se adapta a cualquiera de estas, sin embargo, cuando se pretende ejecutar la inferencia en tiempo real es necesario que el algoritmo esté realizado de manera óptima para que la Intel Neural Computer Stick 2 lo ejecute.

Figura 10

Muestra imagen inferida Raspberry Pi 4 e Intel Neural Stick 2

Intel Movidius acelera la inferencia de redes neuronales y gráficos directos, su ventaja radica en la etapa de preprocesamiento y post procesamiento que reemplaza bucles de programación por capas en los niveles superiores de la red.

La herramienta que permite el acondicionamiento se denomina Model Optimizer

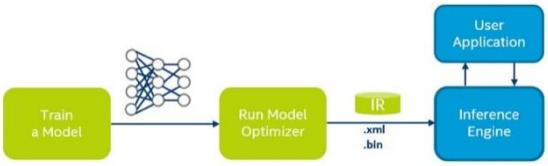
ISSN: **2773-7330**
Vol. 4 No. 3.1 pp. 109 – 126. agosto 2022

www.alfapublicaciones.com

que permite introducir modelos de- sarrollados, por ejemplo, en TensorFlow u OpenCV y permite obtener un modelo útil con extensión .xml y .bin que pueda ejecutarse, el proceso se puede divisar en la figura 11.

Figura 11

Proceso de conversión del modelo



La figura 12 representa una muestra de las imágenes de entrada, la figura 13 es la imagen de salida luego de ser ejecutado el algoritmo en la Raspberry 4 la cual tomó 0.22 segundos en su ejecución.

Figura 12

Imagen original

Fuente: Elaboración propia

Raspberry Pi 4 e Intel Neural Stick 2: Es necesario acondicionar el modelo para ser ejecutado con la Intel Neural Computer Stick 2, para esto el código para ejecutar la inferencia se lo desarrolla en la consola de Python. Se puede distinguir en la figura 14 el resultado. El tiempo de inferencia fue de segundos.

Análisis de resultados

En esta sección se analiza los resultados obtenidos al medir el tiempo de inferencia en los dos algoritmos. Una vez ejecutados sobre las dos combinaciones de hardware antes detalladas se establece dos consideraciones generales:

ISSN: **2773-7330**
Vol. 4 No. 3.1 pp. 109 – 126. agosto 2022

www.alfapublicaciones.com

Figura 13

Imagen de muestra resultante Raspberry Pi 4

Figura 14

Imagen de muestra resultante Raspberry Pi 4 e Intel Neural Stick 2

1) Se toma en cuenta el tiempo medio de clasificación de la Raspberry Pi 4 y la Raspberry Pi 4 con Intel Neural Computer Stick 2.
2) Se aplica una prueba t para medias de dos muestras emparejadas cuya variable uno son los tiempos de inferencia de cada una de las imágenes en la Rapsberry Pi 4 y la variable dos es el tiempo de inferencia en la Raspeberry Pi 4 más la Intel Neural Stick 2.

El análisis estadístico de los resultados parte al establecer los dos posibles escenarios el primero que el dispositivo Raspberry Pi 4 ejecute en menor tiempo la clasificación de imágenes que la Raspberry con Intel Neural Stick, para esto se fija esa hipótesis como hipótesis nula Ho. Los resultados se pueden observar en el cuadro I.

Con un valor en el parámetro α del 5% la diferencia promedio de los pesos, luego de modificar el hardware, es mayor a cero. Es decir que los tiempos de ejecución del

ISSN: **2773-7330**
Vol. 4 No. 3.1 pp. 109 – 126. agosto 2022

www.alfapublicaciones.com

algoritmo de inferencia se reducen en el segundo experimento, por lo tanto, se rechaza la hipótesis nula.

Como se puede divisar en el cuadro II y en la figura 15 los tiempos de respuesta en la Raspberri Pi 4 son más lentos debido al rango intercuartil menor, por lo tanto, el tiempo de respuesta del sistema embebido es menor si no está trabajando en paralelo con el Intel Movidius Neural Compute Stick 2, lo cual le ha permitido alcanzar una mejora del 25%.

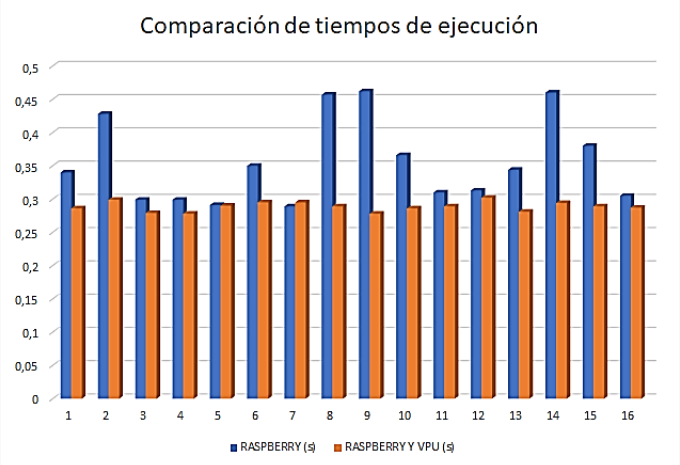
Tabla 1

Estadística de prueba T

	RASPBERRY PI 4	RASPBERRY PI 4 + INTEL NEURAL STICK 2
Media	0,3558125	0,2885625
Varianza	0,004021763	5,28E-05
Observaciones	16	16
Coeficiente de correlación de Pearson	0,061298007	
Diferencia hipotética de las medias	0	
Grados de libertad	15	
Estadístico t	4,243690582	
P(T<=t) una cola	0,000353837	
Valor crítico de t (unas colas)	1,753050356	
P(T<=t) dos colas	0,000707674	
Valor crítico de t (dos colas)	2,131449546	

Figura 15

Tiempos de inferencia para un grupo de imágenes



ISSN: **2773-7330**
Vol. 4 No. 3.1 pp. 109 – 126. agosto 2022

www.alfapublicaciones.com

Tabla 2

Estadística descriptiva de los tiempos de inferencia

	RASPBERRY PI 4	RASPBERRY PI 4 + INTEL NEURAL STICK 2
Media	0,3558125	0,2885625
Error típico	0,01585434	0,001816519
Mediana	0,342	0,289
Moda	0,299	0,289
Desviación estándar	0,06341737	0,007266074
Varianza de la muestra	0,00402176	5,28E-05
Curtosis	-0,93536068	-0,656066026
Coeficiente de asimetría	0,74887509	0,11173525
Rango	0,173	0,024
Mínimo	0,289	0,278
Máximo	0,462	0,302
Suma	5,693	4,617
Cuenta	16	16

Conclusiones

- La primera conclusión gira en torno a los procesos que se llevaron a cabo con el fin de instalar y configurar de manera óptima la Intel Neural Computer Stick 2 de Intel que, al ser una versión nueva, la información disponible por el fabricante era poco útil. También es importante considerar que, en la mayoría de las investigaciones, la combinación de estos dos dispositivos se utilice una Raspberry pi 3 por lo tanto toda la información disponible estaba enfocada a versiones anteriores lo que dificultaba la instalación de librerías fundamentales para que el motor de inferencia funcione.
- Los algoritmos de visión utilizados en este estudio fueron dos elementos fundamentales a considerar, ya que el motor de inferencia era compatible solo con los backends comunes. El motor de inferencia ofrece una herramienta que permite acondicionar modelos entrenados con otros procedimientos y
- optimizarlos para la ejecución en el acelerador. Es importante entender que el proceso de optimización se enfoca en convertir bucles cíclicos en capas de inferencia en la etapa de preprocesamiento.
- Bajo estas condiciones de experimento el porcentaje de tiempo que se reduce en las inferencias aproximadamente es del 25 %, mejorando los tiempos de ejecución. Se debe considerar evaluar el acelerador de Intel con una fuente de

ISSN: **2773-7330**
Vol. 4 No. 3.1 pp. 109 – 126. agosto 2022

imágenes en tiempo real.

Referencias Bibliográficas

Abellán Campos, M. (2021). Implementación y aceleración de algoritmos de IA sobre Raspberry Pi.

Espinola Gonzales, J. E., Asís López, M. E., & Rodríguez Sabino, V. G. (2011). Sistema de visión artificial para la detección de somnolencia de conductores, basado en el comportamiento ocular.

Fernández de la Torre, A. (2020). Portabilidad y optimización de una red neuronal para la detección rápida de daños en terremotos usando el toolkit OpenVINO.

Garduño, J. I., & Padrón, A. (2000). Análisis de Antenas Lineales Empleando Redes Neuronales Artificiales. In X Congreso Internacional de Electrónica, Comunicaciones y Computadoras, CONIELECOMP-2000, Cholula-Puebla, México (pp. 393-397).

Gómez Muñoz, L. F., & Vélez Gutiérrez, J. D. (2015). Simulación del modelo de interacción de actores del sistema de abastecimiento alimentario de Bogotá: implementación de tecnologías de modelado basado en agentes y metodología de sistemas suaves.

Intel.com. (2022). Inferencia de aprendizaje profundo de alto rendimiento. Intel.com. https://www.intel.com/content/www/us/en/developer/tools/openvino-toolkit/overview.html.

Jiménez, M. F. (2012). Redes neuronales y preprocesado de variables para modelos y sensores en bioingeniería (Doctoral dissertation, Universitat Politècnica de València).

Juárez, G., & Pascual, M. (2015). Implementación de redes neuronales sobre lógica reconfigurable. Repositorio Nacional Conacyt.

Matich, D. J. (2001). Redes Neuronales: Conceptos básicos y aplicaciones. Universidad Tecnológica Nacional, México, 41, 12-16.

Montoya Vásquez, D. A. (2020). Implementación y evaluación del rendimiento de redes neuronales densas en FPGA para la inferencia rápida, aplicadas a problemas en física y visión artificial.

Ordoñez Ramos, E. (2020). Deep Learning para la visión artificial e identificación del personal administrativo y docente de la Universidad Nacional Micaela Bastidas

de Apurímac 2018.

Raspberry Pi. (2022). Computing for everybody. raspberrypi.com. https://www.raspberrypi.com/

Rengifo Mora, F. A. (2002). Determinación de fases sísmicas PN usando redes neuronales.

Rodríguez, S. A., & Ruz Gómez, R. (2020). Evaluación de algoritmos de machine learning para conducción.

Salazar Vásquez, F. A. (2017). Implementación de un algoritmo de aprendizaje para la arquitectura Deep-Networks.

Vázquez Valle, G. (2020). Diseño de un sistema basado en redes neuronales para la detección de tumores cerebrales mediante imágenes hiperespectrales.

ISSN: **2773-7330**
Vol. 4 No. 3.1 pp. 109 – 126. agosto 2022

www.alfapublicaciones.com

El artículo que se publica es de exclusiva responsabilidad de los autores y no necesariamente reflejan el pensamiento de la **Revista Alfa Publicaciones.**

El artículo queda en propiedad de la revista y, por tanto, su publicación parcial y/o total en otro medio tiene que ser autorizado por el director de la **Revista Alfa Publicaciones.**