DISCA

DEPARTAMENTO DE INFORMÁTICA DE SISTEMAS Y COMPUTADORES

Actas de la Jornada de Arquitectura y Tecnología de Computadores 2024

Edita:
Departamento de Informática de Sistemas y Computadores (DISCA)
Valencia

DISCA
DEPARTAMENTO DE INFORMÁTICA DE SISTEMAS Y COMPUTADORES

Actas de la Jornada de Arquitectura y Tecnología de Computadores 2024

Valencia, 31 de enero de 2024

Número: 1
ISSN: 3020-9943
Edita: Departamento de Informática de Sistemas y Computadores (DISCA) – Universitat Politècnica de València

Índice:

Análisis de interferencias, escalabilidad de las prestaciones y consumo energético en un ARM ThunderX2

Ibai Calero[1], Salvador Petit[1], María E. Gómez[1] y Julio Sahuquillo [1],

Resumen— **Hoy en día la eficiencia energética es fundamental en todo tipo de dispositivos, desde servidores a supercomputadores, pasando por dispositivos alimentados por batería. El diseño de software y hardware energéticamente eficiente requiere del estudio de la relación entre 3 ejes principales: actividad de los componentes, consumo e interferencias entre aplicaciones en los recursos compartidos. Este estudio caracteriza los ejes mencionados anteriormente en un procesador ARM ThunderX2. Concluyendo con que las prestaciones de las aplicaciones mono-hilo con poca sensibilidad a las interferencias se sostienen sin importar el número de aplicaciones que se ejecuten simultáneamente. En cambio, aplicaciones con una alta sensibilidad presentan un consumo menor pero experimentan elevadas degradaciones de prestaciones al sufrir interferencias.**

Palabras clave— **ARM ThunderX2, Caracterización, prestaciones, consumo energético, interferencias entre aplicaciones.**

I. Introducción y Motivación

Las transiciones digital y energética son esenciales para lograr una economía sostenible y constituyen un pilar fundamental de la nueva estrategia industrial a nivel global. Sin embargo, el aumento vertiginoso de la cantidad de datos y la demanda de servicios soportados por los Centros de Datos (CD) podría llevar a un incremento exponencial en el consumo de energía. Requiriendo diseñar sistemas de hardware y software energéticamente eficientes para abordar este crecimiento.

El procesador es el consumidor de energía dominante en los CD, siendo responsable del 61 % del consumo [1]. Para lidiar con esto, algunos estudios como Bertran et al. [2] se centran en abordar los desafíos de la monitorización energética en entornos virtualizados. Otros como Isci et al. [3] se enfocan en modelar el consumo de energía dentro del procesador desglosado en muchos componentes arquitectónicos (e.g. el *reorder buffer* o el *register file*), mientras que Chen et al. [4] intentan lograr una estimación precisa y en tiempo real del rendimiento y el consumo de energía en escenarios donde múltiples procesos interactúan en un procesador de cuatro núcleos.

Este estudio examina la sensibilidad de las prestaciones y el consumo a las interferencias en un procesador ARM ThunderX2 de 28 núcleos, analizando también la relación entre la actividad de los componentes, las interferencias de prestaciones y el consumo de energía. Se identifican tres tipos de aplicaciones: limitadas por CPU, memoria y caché L3. Estas aplicaciones también se clasifican en escalabilidad alta, media o baja según cómo se comportan sus prestaciones frente a interferencias causadas por otras aplicaciones. Los resultados indican que las prestaciones de aplicaciones altamente escalables apenas se ven afectadas por las interferencias, pero experimentan un aumento significativo en el consumo de energía. Por el contrario, ejecutar muchas aplicaciones de baja escalabilidad reduce el consumo de energía, aunque con una caída notable en las prestaciones.

[1]DISCA, Universitat Politècnica de València, e-mails: icalqui@inf.upv.es {spetit,megomez,jsahuqui}@disca.upv.es

II. Sistema Experimental y Metodología

Se ha utilizado un equipo con CentOS Linux 7 y la versión 4.18.0 del kernel de Linux. El procesador empleado es un ARM ThunderX2 con 28 núcleos y 64 GB de memoria DDR4 a 2666 MHz. Dispone además de 3 niveles de caché. Solo el tercero es compartido por todos los núcleos y tiene una capacidad de 32 MiB. La frecuencia de los núcleos se ha fijado a 2,5 GHz (la máxima soportada). El procesador dispone de 4 contadores de energía que monitorizan el consumo de los dominios principales del procesador (cores, L3, SoC y componentes misceláneos del SoC). Estos contadores miden la energía estática (consumida solo por estar funcionando) y la dinámica (causada por las aplicaciones).

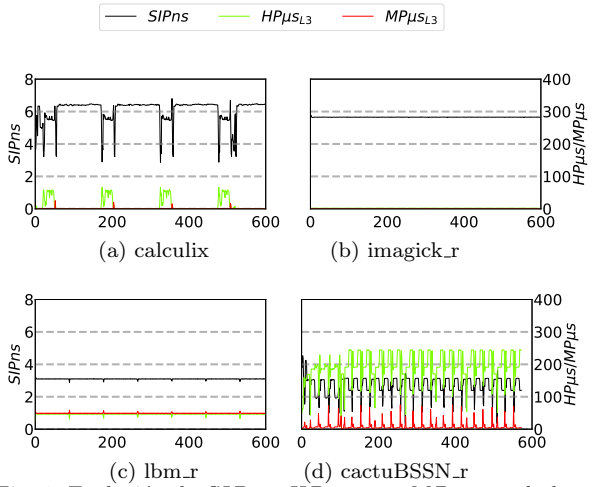
En la caracterización, el impacto de la energía estática se ha mitigado estimándola (consumo del sistema cuando no se ejecuta ninguna aplicación) y restándola a la energía medida por los contadores.

Como benchmarks hemos utilizado aplicaciones de las suites SPEC CPU 2006 y 2017, aunque en este estudio solo mencionamos un subconjunto de ellas.

III. Caracterización Individual

Hemos identificado métricas correlacionadas con el trabajo realizado por los componentes del procesador que más energía consumen: los núcleos y la caché L3. Las instrucciones especuladas por nanosegundo ($SIPns$) para los núcleos y para la caché L3 sus aciertos y fallos por microsegundo ($HP\mu s_{L3}$ y $MP\mu s_{L3}$, respectivamente).

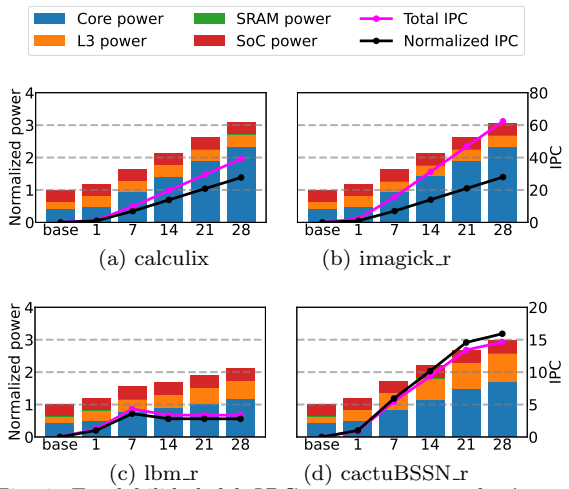
En base a estas métricas (mostradas en la Figura 1), hemos identificado 3 tipos de comportamientos.
Limitadas por CPU. Integrada por las aplicaciones que presentan una elevada actividad en la CPU ($SIPns$ por encima de 5) y pocos accesos a la caché L3 ($HP\mu s_{L3}$ y $MP\mu s_{L3}$ cercanos a 0). Es el caso de *calculix* e *imagick_r*.



(a) calculix (b) imagick_r

(c) lbm_r (d) cactuBSSN_r

Fig. 1: Evolución de $SIPns$, $HP\mu s_{L3}$, y $MP\mu s_{L3}$ a lo largo del tiempo.



(a) calculix (b) imagick_r

(c) lbm_r (d) cactuBSSN_r

Fig. 2: Escalabilidad del IPC y consumo con el número de instancias.

Limitadas por memoria. Compuesta por aplicaciones con una elevada tasa de fallos en la caché L3 ($MP\mu s_{L3}$) y, por tanto, muchos accesos a memoria principal. Es el comportamiento de *lbm_r*.

Limitadas por la L3. Incluye aplicaciones con una elevada tasa de aciertos en la caché L3 (más de 100 aciertos por μ) y un bajo número de fallos. Es el comportamiento presentado por *cactuBSSN_r*.

Aunque omitido por limitaciones de espacio, se ha comprobado cómo a mayor número de $SIPns$ mayor es el consumo de los núcleos. Análogamente, cuanto mayores son las tasas de aciertos y fallos de la caché L3 mayores son los consumos de esta.

IV. Sensibilidad a las interferencias

A continuación se ha estudiado el efecto en las prestaciones y el consumo de energía que tiene aumentar el número de instancias ejecutándose simultáneamente de 1 hasta 28 (número de núcleos).

La figura 2 muestra como las prestaciones (IPC) y la energía crecen con el número de instancias. Se utiliza el IPC, tanto el total (suma del de cada instancia) como el normalizado (respecto al de 1 instancia), para evaluar las prestaciones. El consumo de energía está dividido entre los distintos dominios del procesador y normalizado respecto al consumo del sistema cuando no se ejecutan aplicaciones.

Desde la perspectiva de cómo escala el rendimiento con el número de instancias, las aplicaciones se pueden clasificar en 3 categorías.

Escalabilidad elevada. Es el comportamiento de todas las aplicaciones limitadas por CPU. Las prestaciones crecen prácticamente linealmente con el número de instancias (llegando a un IPC normalizado de 28 para 28 instancias). En consecuencia, el consumo también se incrementa de manera lineal con el número de instancias, con los núcleos siendo el componente principal en el consumo.

Escalabilidad intermedia. Es el caso de *cactuBSSN_r* (limitada por la L3). Esta presenta un crecimiento menos agresivo en sus prestaciones con el número de instancias. Esto se traduce en un consumo de los núcleos menor que el de las aplicaciones con una escalabilidad elevada.

Escalabilidad reducida. Únicamente *lbm_r* (limitada por memoria) presenta este comportamiento. Sus prestaciones presentan un escaso crecimiento, llegado a experimentar una reducción al incrementar el número de instancias de 7 a 14. Presenta además los consumos más reducidos de las aplicaciones estudiadas.

V. Conclusiones

Hemos evaluado aplicaciones de un solo hilo de las suites SPEC CPU 2006 y 2017 en un procesador ARM ThunderX2 desde la perspectiva del consumo energético. Identificamos 3 tipos de aplicaciones según el componente que más estresan: limitadas por CPU, memoria y la caché L3. Además, analizamos cómo evolucionan sus prestaciones al aumentar las interferencias, distinguiendo entre alta, media y baja escalabilidad. Las aplicaciones con alta escalabilidad muestran consumos y prestaciones elevadas, que aumentan linealmente con el número de instancias. Por otro lado, las aplicaciones con una escalabilidad media y baja degradan sus prestaciones al aumentar las interferencias, pero también presentan un menor consumo de energía.

Agradecimientos

Este trabajo ha sido parcialmente financiado por el Ministerio de Ciencia e Innovación de España y el FEDER europeo a través de las subvenciones PID2021-123627OB-C51 y TED2021-130233B-C32.

Referencias

[1] L. A. Barroso *et al.*, *The Datacenter as a Computer: Designing Warehouse-Scale Machines, Third Edition*, Synthesis Lectures on Computer Architecture. Morgan & Claypool Publishers, 2018.

[2] R. Bertran *et al.*, "Energy accounting for shared virtualized environments under dvfs using pmc-based power models," *Future Generation Computer Systems*, vol. 28, no. 2, pp. 457–468, 2012.

[3] C. Isci and M. Martonosi, "Runtime power monitoring in high-end processors: methodology and empirical data," in *Proc of MICRO*, 2003, pp. 93–104.

[4] X. Chen *et al.*, "Performance and power modeling in a multi-programmed multi-core environment," in *Proc of DAC*, 2010, p. 813–818.

Monitorización eficiente de Ramblas con *BODOQUE*

Benjamín Arratia[1], Erika Rosas, Salvador Peña-Haro, José M. Cecilia y Pietro Manzoni

Resumen— **El monitoreo eficiente de corrientes efímeras enfrenta desafíos únicos debido a su naturaleza impredecible y esporádica. Este trabajo introduce *BODOQUE*, un sistema que combina TinyML con técnicas avanzadas de monitoreo para mejorar la precisión y eficiencia energética en la medición de estas corrientes. Destacamos la introducción de un nuevo enfoque para la selección de modelos TinyML, centrado en optimizar tanto el rendimiento como el costo de inferencia, y el desarrollo de un dispositivo especializado para la recopilación de datos que facilita la mejora continua del sistema.**

Palabras clave— **Internet of Things, Tiny Machine Learning, Edge Computing, Environmental Intelligence, Energy Efficiency, Adaptive Systems, Streamflow Monitoring**

I. Introducción

El estudio y monitorización de corrientes efímeras son de gran importancia en los campos de la ecología y la hidrología, especialmente frente a los desafíos planteados por el cambio climático y la gestión sostenible de recursos hídricos. Tradicionalmente, la medición de estos flujos ha requerido de sistemas de monitorización continua que, a pesar de su eficacia, incurren en un alto consumo energético y operativo. *BODOQUE* se presenta como una solución que integra *Machine Learning* en microcontroladores (TinyML) de bajo consumo para activar un sistema de procesamiento de mayor consumo energético solo cuando es necesario. Este sistema no solo promete mejorar la eficiencia operativa y la precisión en la detección de eventos hídricos, sino también reducir de manera significativa el impacto ambiental asociado al consumo energético de estos sistemas de monitoreo.

II. Antecedentes

El estudio de corrientes efímeras es crucial para la gestión sostenible de recursos hídricos y la comprensión de ciclos ecológicos. Dado que estos arroyos son intermitentes y están en ubicaciones remotas, monitorearlos eficazmente es un desafío [1]. Los métodos convencionales son precisos, pero energéticamente intensivos, lo que los hace imprácticos para monitoreos prolongados en zonas inaccesibles. Las técnicas modernas de procesamiento de imágenes y aprendizaje automático han permitido superar estas barreras, proporcionando mediciones precisas del flujo de agua con menor consumo energético [2].

TinyML emerge como solución para estos retos, integrando modelos de aprendizaje automático en microcontroladores de bajo consumo [3]. Al procesar datos localmente (*Edge Computing*), reduce la necesidad de transmisión continua de datos, disminuyendo el consumo energético. Los sistemas de monitoreo ambiental basados en TinyML ofrecen flexibilidad y adaptabilidad, ajustándose en tiempo real a cambios ambientales [4].

La implementación de TinyML en la monitorización de corrientes efímeras promete eficiencia y sostenibilidad. Sin embargo, elegir y optimizar un modelo TinyML adecuado es un desafío, ya que debe equilibrar precisión de detección, eficiencia operativa y consumo de recursos. Proponemos un enfoque sistemático para optimizar la selección de modelos TinyML, buscando maximizar la eficiencia energética sin sacrificar la calidad de los datos recopilados.

III. El Sistema *BODOQUE*

BODOQUE (Por sus siglas en inglés **B**imodal **O**bservational **D**evice for **O**ptimizing **QU**antification of **E**phemeral streams) [5] se caracteriza por su estructura de doble módulo, diseñado para alternar entre una fase de detección de bajo consumo y una fase de medición de alta precisión al detectar la presencia de agua en el cauce. Este enfoque permite optimizar el consumo energético al mantener el sistema en un estado de bajo consumo hasta que es necesario realizar mediciones precisas.

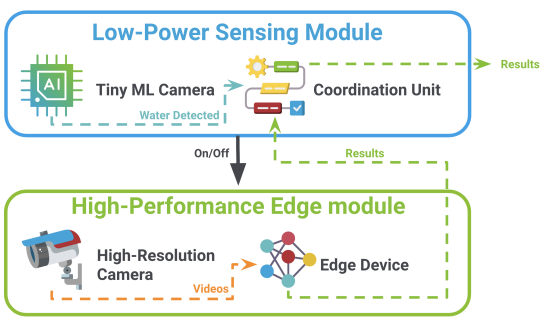


Fig. 1: Módulos *BODOQUE*

El Módulo de Detección de Bajo Consumo (LPSM) utiliza una cámara equipada con capacidades TinyML para monitorear continuamente el entorno en busca de cambios, mientras que el Módulo de Alto Rendimiento (HPEM) se activa únicamente cuando se detecta agua, realizando mediciones detalladas del caudal mediante técnicas avanzadas de procesamiento de imágenes. Una vez el caudal termina, el HPEM envía una señal al LPSM, para terminar con el procesamiento y comenzar nuevamente el proceso de detección de agua de bajo consumo energético. Este diseño no solo mejora la eficiencia energética, sino que también garantiza la precisión y

[1]DISCA, Universitat Politècnica de València.
e-mail: baarruri@upv.edu.es.

fiabilidad de las mediciones en condiciones ambientales variables.

IV. TinyML en *BODOQUE*

La eficiencia en la monitorización de corrientes efímeras mediante *BODOQUE* depende críticamente de la capacidad para detectar la presencia de agua a tiempo. En este contexto, resulta esencial elegir correctamente el modelo de TinyML. Para ello, hemos desarrollado *MeTINCA* (**M**odel **E**fficiency **T**rade-off and **In**ference **C**ost **A**nalysis), una metodología que busca equilibrar la eficiencia operativa con el costo de inferencia, priorizando modelos que combinen alta precisión en la detección con un bajo consumo energético.

Hemos entrenado una variedad de modelos utilizando redes neuronales convolucionales y evaluado su desempeño en hardware específico, con el objetivo de optimizar el valor de la métrica *MeTINCA*. El modelo seleccionado, que implementa una arquitectura Conv2D de doble capa, se distingue por su precisión en detectar agua, así como por sus rápidos tiempos de inferencia y el uso moderado de RAM y ROM.

Las imágenes para el entrenamiento y validación del modelo se obtienen inicialmente de una cámara de alta resolución, lo que garantiza una captura detallada de las corrientes bajo variados escenarios ambientales. Para las futuras iteraciones y reentrenamientos, utilizaremos el *BODOQUE Dataset Collector*, un dispositivo diseñado para la recolección y etiquetado automático de nuevas imágenes en el terreno. Esta estrategia nos permitirá mejorar continuamente el dataset, ampliando la diversidad de condiciones ambientales representadas y aumentando la robustez del modelo.

V. Experimentos y Resultados

La evaluación del sistema *BODOQUE* se basó en una comparación teórica utilizando datos recopilados de una rambla durante el año 2023. Esta comparativa se centró en el consumo energético del sistema actualmente desplegado en el terreno frente al consumo energético proyectado para *BODOQUE*, considerando su diseño y las especificaciones técnicas propuestas.

La simulación utilizó datos de la Rambla del Albujón", una corriente efímera que experimenta flujos de agua de manera esporádica y representa los desafíos de monitoreo en estas condiciones. A través de este estudio comparativo, se pudo estimar el rendimiento de *BODOQUE* en un escenario real, proveyendo una base sólida para la justificación de su futura implementación.

Los datos analizados revelaron que el sistema *BODOQUE*, con su enfoque optimizado e integración de TinyML, podría lograr una reducción del 97 % del consumo energético en comparación con el sistema de monitoreo existente. Este resultado subraya la eficiencia energética inherente al diseño de *BODOQUE*.

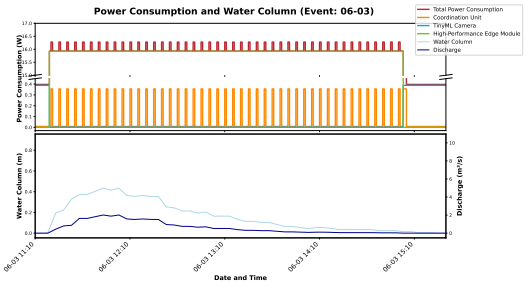


Fig. 2: Comportamiento durante corriente efímera del 2023-06-03

VI. Conclusiones

Este estudio demuestra el potencial de *BODOQUE* para mejorar la monitorización de corrientes efímeras en áreas remotas, proyectando una reducción del consumo energético de hasta un 97 % en comparación con sistemas convencionales de cámaras. Aunque pendiente de implementación en terreno, la simulación basada en datos reales de la Rambla del Albujón"valida la viabilidad y efectividad de *BODOQUE*, respaldando su promesa de un monitoreo ambiental más sostenible y eficiente.

La futura puesta en marcha de *BODOQUE* en entornos reales no solo permitirá validar estos resultados teóricos, sino también afinar el sistema mediante la retroalimentación continua y la expansión del dataset a través del *BODOQUE Dataset Collector*. Estas acciones consolidarán la base para una monitorización ambiental avanzada, marcando un paso significativo hacia la sostenibilidad y precisión en la gestión de recursos hídricos.

Agradecimientos

Este trabajo ha sido apoyado por el programa de investigación e innovación Horizonte 2020 de la Unión Europea bajo el acuerdo de subvención No 101017861 y también por el proyecto Ramón y Cajal Grant RYC2018-025580-I, financiado por MCIN/AEI/ 10.13039/501100011033 y por ESF Invertir en tu futuro, GVA GRISOLIAP/2021/103 y la Beca TED2021-130890B financiada por MCIN/AEI y por la Unión Europea NextGenerationEU/PRTR.

Referencias

[1] Javier Senent-Aparicio, Adrián López-Ballesteros, Anders Nielsen, and Dennis Trolle, "A holistic approach for determining the hydrology of the mar menor coastal lagoon by combining hydrological & hydrodynamic models," *Journal of Hydrology*, vol. 603, pp. 127150, 2021.

[2] Salvador Peña-Haro, Maxence Carrel, Beat Lüthi, Issa Hansen, and Robert Lukes, "Robust image-based streamflow measurements for real-time continuous monitoring," *Frontiers in Water*, p. 175, 2021.

[3] Matthew Cowan, Thierry Moreau, Tianqi Chen, James Bornholt, and Luis Ceze, "Automatic generation of high-performance quantized machine learning kernels," in *Proceedings of the 18th ACM/IEEE Symposium on Code Generation and Optimization*. ACM, 2020, pp. 242–253.

[4] Pete Warden and Daniel Situnayake, *TinyML: Machine learning with tensorflow lite on arduino and ultra-low-power microcontrollers*, O'Reilly Media, 2019.

[5] Benjamín Arratia, Erika Rosas, Carlos T. Calafate, Juan-Carlos Cano, José M. Cecilia, and Pietro Manzoni, "AlLoRa: Empowering environmental intelligence through an advanced LoRa-based IoT solution," *Computer Communications*, vol. 218, pp. 44–58, 2024.

Desarrollo y análisis de redes neuronales convolucionales CNN para la realización de análisis polínicos en la miel

Juan José Martín Osuna[1], José Miguel Valiente[1], Fernando López[1], Isabel Escriche[2], Marisol Juan-Borrás[2] y Mario Visquert-Fas[2]

Resumen— **En este articulo se texto se analizara la topología de las redes Polenet V.2. y Polenet V.2.Mobile, así como, se compraran sus rendimientos con otras redes CNN de propósito general, para la tarea de clasificación de variedades de polen.**

Palabras clave— **Análisis Polínico, Redes Neuronales Convolucionales, conjuntos de muestras de pólenes, Visión por Computador**

I. Introducción

EL análisis polínico representa un procedimiento fundamental para elucidar la monofloralidad de la miel. La metodología estandarizada implica la preparación de la muestra para su examen microscópico, seguido por el recuento manual de 500 a 600 granos de polen. Posteriormente, se determinan los porcentajes relativos de cada tipo de polen presente, lo que permite atribuir una variedad botánica específica a la miel en cuestión.

Este procedimiento es tedioso y complejo, por lo que se desarrolló en el contexto de los proyectos de investigación AGROMEL (1) y POLENET (2) la herramienta **HoneyApp** [1], que permite el etiquetado de las imágenes de polen obtenidas con el microscopio óptico.

Una vez desarrollada e implementada la herramienta, se ha podido obtener un número de muestras tal, con el que proceder a entrenar redes CNN para la clasificación. En este artículo se compararán las topologías más relevantes en el contexto de las CNN, así como se profundizará en las dos nuevas topologías específicas desarrolladas dentro del proyecto POLENET [2].

II. Material y métodos

El entrenamiento de las redes se ha realizado empleando un conjunto de muestras propio, que cuenta con un total de 32,297 muestras, divididas entre 24, siendo estas las que se pueden apreciar en la Figura 1. Estos 21 tipos corresponden a algunas de las variedades de polen más comunes en las mieles del territorio español.

Cabe destacar que los 21 tipos de polen permiten la determinación de hasta 11 variedades de miel monofloral, tales como: Castaño, Azahar, Chupamieles, Brezo, Eucalipto, Girasol, Cantueso, Esparceta, Almendro, Romero y Tomillo.

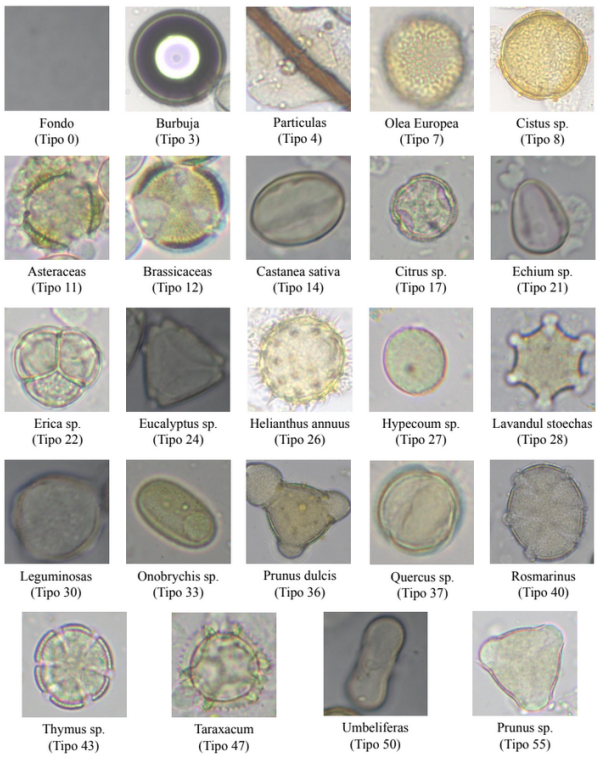


Fig. 1: Imagen representativa de los 24 tipos presentes en el conjunto de muestras de pólenes empleado.

Para abordar el aspecto de las redes, se ha llevado a cabo una selección de doce topologías muy diversas. Estas engloban modelos ampliamente reconocidos en el ámbito de la clasificación de imágenes, tales como: VGG16, VGG19, DenseNet201, MobileNetV2, ResNet50, InceptionV3, Xception, EfficientNetV2M, NASNet, APFANet [3] o las dos versiones más recientes de la red propia del proyecto POLENET: la PolenetV.2 y PolenetV.2 Mobile.

III. Análisis de las topologías de las redes Polenet

La red PolenetV.1 es una red de estructura lineal, muy similar a las redes VGG, pero en este caso con solo cinco capas convolucionales. Tal como se describe tanto en el artículo [1] como en el proyecto [2]. Esta topología de red le confiere una exactitud considerable con un tamaño muy contenido, en comparación con redes de propósito general.

Con esta estructura en mente, se puede apreciar

[1]Instituto de Automática e Informática Industrial, UPV Valencia, España
[2]Instituto Universitario de Ingeniería de Alimentos-FoodUPV, UPV Valencia, España

cómo la red PolenetV.2 y la PolenetV.2 Mobile son evoluciones directas de esta arquitectura original. Por un lado, la red PolenetV.2 presenta la misma distribución de capas que la red original, pero empleando capas de "Batch Normalization", entre la capa convolucional y la de "Max Pooling". Este tipo de capas acelera el entrenamiento y reduce el riesgo de sobreentrenamiento. Con este planteamiento, la estructura queda tal como se puede apreciar en la figura 2.

Por otro lado, la red PolenetV.2 Mobile parte de los cambios implementados en la red PolenetV.2, pero pretende ser una versión más ligera de la misma, sustituyendo la consecución de capas tipo "Dense"de la salida por una capa de "GlobalAvgPooling", seguida de una capa densa con muchos menos parámetros. Quedando la estructura de la red tal y como se describe en la figura 3.

IV. Resultados experimentales

Una vez prestadas las redes desarrolladas en el proyecto, se procederá a realizar un entrenamiento de 120 épocas para cada una de las redes seleccionadas para la comparación. Todos los entrenamientos se harán con el mismo dataset, el desarrollado en el contexto del proyecto con los 24 tipos distintos y en todos los casos el dataset estará dividido de la misma manera, siendo el 80 % de las imágenes empleadas para el entrenamiento, el 10 % para el test y el último 10 % para la validación.

Descritas las condiciones del entrenamiento, se ha podido obtener la Tabla I, que permite comparar los resultados de cada red en los parámetros habituales, así como el tamaño en MB.

Tabla I: Resultados comparativos que relaciona la exactitud, precisión, Recall y tamaño de las distintas arquitecturas.

	Exactitud(%)	**Precisión (%)**	**Recall (%)**	**Tamaño (MB)**
VGG16	97.42	97.00	97.40	156.4
VGG19	97.45	97.40	97.40	177.1
ResNet50	97.62	97.60	97.60	485.5
InceptionV3	97.98	98.00	98.00	598.8
Xception	97.75	97.80	97.80	882.7
DenseNet201	97.83	97.80	97.80	441.8
EfficientNetV2M	98.03	98.00	98.00	530.4
MobileNetV2	96.30	96.60	96.40	255.1
NASNet	97.61	98.00	97.80	24.1
APFANet	96.87	97.00	97.00	15.3
PolenetV.2	97.10	96.80	96.80	144.2
PolenetV.2.Mobile	96.48	96.40	96.40	16.4

V. Conclusiones

A la vista de los resultados, se puede apreciar que por lo que respecta a los porcentajes de Exactitud, Precisión y Recall, todas las redes se encuentran en en un rango de 97 %, con una desviación estándar del ±2. Debido a que la clasificación de las variedades de polen es una tarea poco crítica, se puede concluir que todas las redes serian aptas para su implementación en la herramienta.

Por lo tanto, el factor más determinante a la hora de seleccionar una red para su implementación en la herramienta es su tamaño. Dentro de este campo se puede concluir que las redes que mejor relación tamaño/exactitud son la PolenetV.2.Mobile y la APFANet, las cuales son redes desarrolladas específicamente con este propósito. Por otro lado aunque la red Polenet V.2. no tiene tanta buena relación tamaño/exactitud, tiene unos valores de exactitud más cercanos a las mejores redes analizas, como son la EfficientNetV2M o la InceptionV3, pero con un tamaño de casi 4 ordenes de magnitud menor.

Agradecimientos

(1) Proyecto AGROMEL-AGROALNEXT/2022/043, "Técnicas analíticas rápidas para evaluar seguridad, adulteración y trazabilidad en productos de la colmena. Aplicación a un cultivo en transición agroecológica", Generalitat Valenciana. Next Generation European Union y PERTE de España

(2) Proyecto POLENET PID2019-106800RB-I00 (2019) del Ministerio de Ciencia e Innovación (MCIN), Agencia Estatal de Investigación MCIN /AEI/10.13039/501100011033.

Referencias

[1] J. M. Valiente, *Automatic pollen recognition using convolutional neural networks: The case of the main pollens presents in Spanish citrus and rosemary honey.*, Journal of Food Composition and Analysis, 123, 105605., 2023.

[2] J. J. Martín Osuna, *Desarrollo de una red neuronal convolucional (CNN) para la distinción de variedades de pólenes en muestras de miel.*, RiuNet(UPV), 2024.

[3] T. Mahmood, *Artificial intelligence-based classification of pollen grains using attention-guided pollen features aggregation network.*, Journal of King Saud University-Computer and Information Sciences, 35(2), 740-756., 2023.

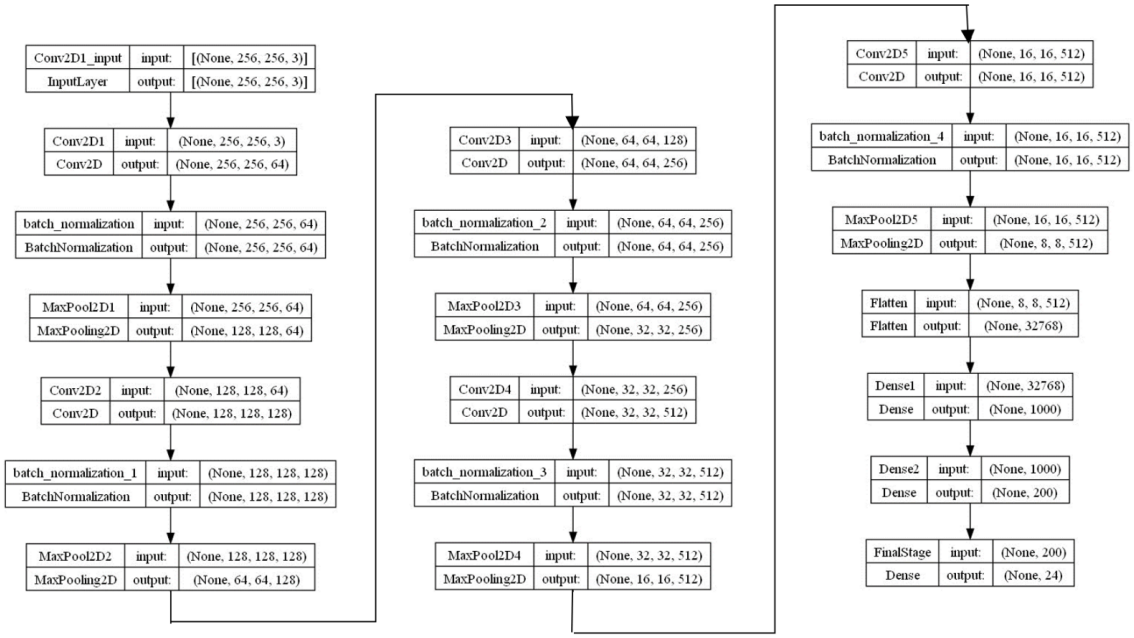


Fig. 2: Imagen representativa de la arquitectura de la red PolenetV.2.

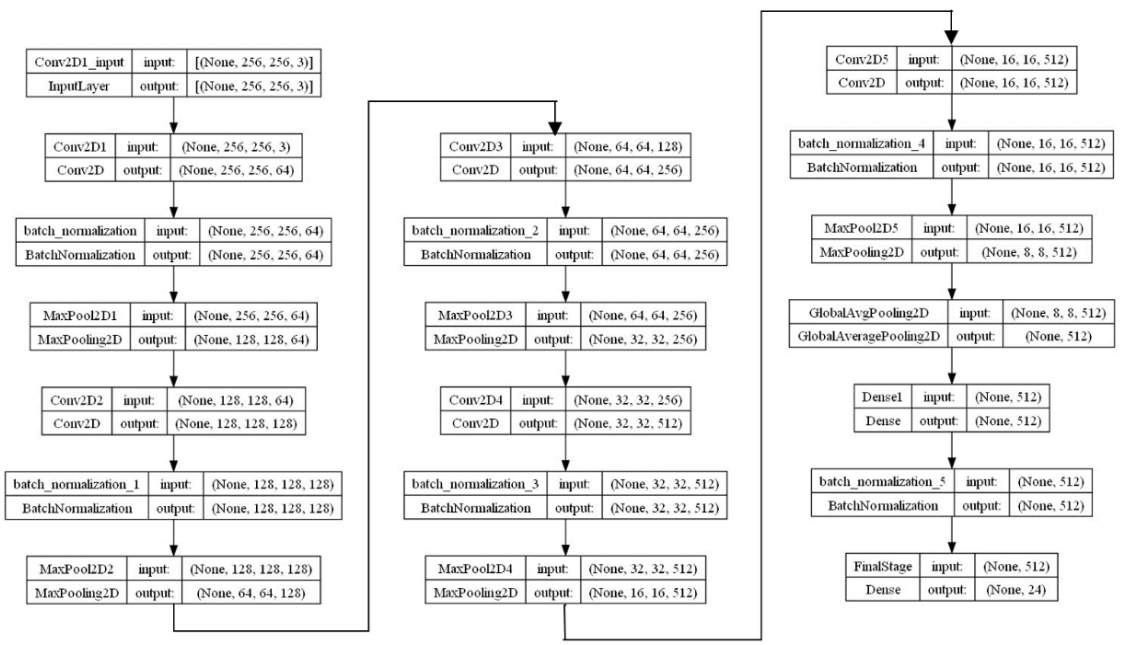


Fig. 3: Imagen representativa de la arquitectura de la red PolenetV.2 Mobile.

Estudio sobre la redundancia a nivel de bit y el impacto de Bit Flips en las Redes Neuronales Convolucionales

Izan Catalán Gallach[1], José Flich Cardo[1] y Carles Hernández Luz[1]

Resumen— **Las redes neuronales convolucionales se están utilizando masivamente en entornos críticos como la salud, los vehículos autónomos o la videovigilancia. Por tanto, es esencial validar su correcto funcionamiento para proporcionar seguridad a los sistemas que las utilizan. En este trabajo analizamos el comportamiento de varios modelos de Redes Neuronales Convolucionales (CNNs) en la presencia de fallos. Para ello, estudiamos y analizamos la sensibilidad de estos modelos e identificamos el impacto de los cambios de bit en su precisión así como el origen de los mismos.**

Palabras clave— **Redes Neuronales, Convoluciones, Invariantes**

I. Introducción

Las redes neuronales convolucionales (CNN) [1] han demostrado ser altamente efectivas en el procesamiento de imágenes [2], la conducción autónoma [3] y otros campos debido a su capacidad para manejar datos visuales [4]. Estas redes utilizan técnicas como la cuantización, *tailoring* [5] y *sparsity* [6] para mejorar el rendimiento de la inferencia.

Las CNN consisten principalmente en capas de convolución, *pooling* y *fully connected*. Las capas de convolución aplican filtros a las imágenes de entrada, las capas de *pooling* reducen la dimensionalidad de las salidas de la convolución y las capas *fully connected* asignan las características extraídas a las clases de salida.

Estas redes se utilizan en sistemas críticos donde la detección, segmentación y clasificación de imágenes son esenciales, y deben estar protegidas contra fallos. Tras el entrenamiento y optimización, el rendimiento de las CNN puede verse afectado por fallos en producción debido a factores ambientales o internos, como la corrupción de memoria o la degradación por uso. Estos fallos pueden alterar los parámetros o estados intermedios de las redes [7] [8], llevando a predicciones incorrectas con posibles consecuencias graves [9], como confundir un animal con un coche [10][11][12].

La redundancia a nivel de bit en las CNN, debido a la organización de los pesos dentro del umbral [-1,1] en modelos FP32, es un área de interés. Este estudio busca verificar esta redundancia para desarrollar mecanismos de tolerancia a fallos.

Además, se analiza la robustez de las CNN frente a fallos mediante inyecciones de *bit flips* en varios modelos, incluyendo el impacto de la precisión reducida como INT8. Los resultados indican que la precisión Top1 de los modelos FP32 disminuye significativamente, del 1,3 % al 3 %, con la inyección de un solo *bit flip*.

El documento se estructura de la siguiente manera: la sección II revisa los trabajos relacionados con la protección de las CNN, la sección III presenta un análisis bit-a-bit de los modelos CNN, la sección IV detalla los experimentos y resultados obtenidos, y la sección V concluye el estudio.

II. Estudios de Protección sobre las CNN

En los últimos años se han desarrollado diversas soluciones para mejorar la robustez de las CNN ante fallos, abarcando tanto la detección como la corrección de errores.

Una solución común es la redundancia, como la modular doble o triple, aunque es costosa. Alternativamente, se pueden proteger componentes específicos de hardware o utilizar códigos de corrección de errores (ECC) para la memoria [13]. También se ha investigado la resiliencia a errores en aceleradores de redes neuronales a nivel de transistor y con memoria 3D *die-stacked* [14].

A nivel de software, se pueden añadir restricciones durante el entrenamiento para aumentar la resiliencia o reconstruir los pesos durante la inferencia [15]. Sin embargo, estos métodos pueden reducir la precisión de la red. Otra técnica es entrenar una red neuronal secundaria para verificar la salida de la principal, aunque esto requiere recursos adicionales y puede reducir la precisión [16].

El uso de *checksums* para proteger grupos de pesos también ha sido explorado, siendo más efectivo en grupos pequeños [17]. Además, se han aplicado funciones *hash* a capas específicas para detectar errores, aunque no los corrigen. Finalmente, se han implementado mecanismos de protección basados en umbrales para limitar el impacto de los cambios de bit [18].

III. Análisis bit-a-bit de modelos de Redes Neuronales Convolucionales

En esta sección se examinan los pesos de varios modelos de CNN para identificar patrones bit-a-bit, centrándose en posibles redundancias en el exponente de números decimales en FP32.

Analizamos los modelos MobileNet, ResNet50, VGG16, DenseNet y ShuffleNet, tanto en FP32 como en INT8 (ver Tabla I). Los modelos se descargaron del repositorio ONNX Model Zoo [19]. Se agruparon

[1]Dpto. de Informática de Sistemas y Computadores, Universitat Politècnica de València, e-mail: izcagal@inf.upv.es, jflich@disca.upv.es y carherlu@upv.es

Tabla I: Especificaciones de los Modelos de Redes Neuronales Convolucionales.

Model	Data Type	Top 1 Acc %	Top 5 Acc %	Data Type	Top 1 Acc %	Top 5 Acc %
MobileNet v2-1.0	FP32	69.45	89.23	INT8	62.43	84.54
ResNet-50 v2-1.0	FP32	75.01	92.38	INT8	69.00	88.63
VGG-16	FP32	72.38	91.02	INT8	69.30	88.83
DenseNet-121-12	FP32	60.94	84.49	INT8	59.92	83.64
ShuffleNet-v2	FP32	66.38	86.58	INT8	58.60	80.40

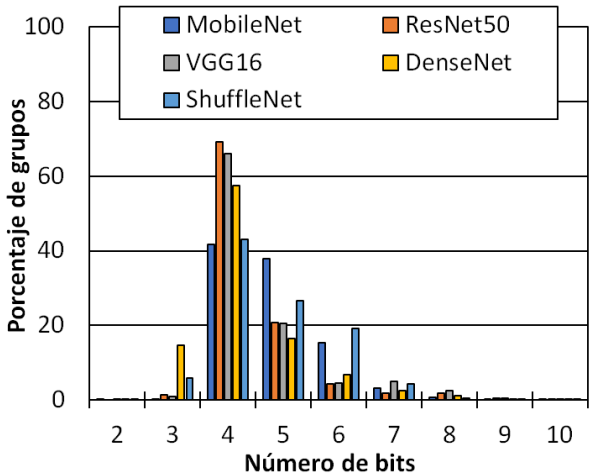


Fig. 1: Número de bits invariantes por grupos (FP32)

los pesos de las capas de convolución en conjuntos de 9 pesos. Para filtros de 3×3, todos los pesos de un filtro se agrupan juntos, y para filtros de 1×1, se agrupan nueve filtros.

Se analizó la redundancia de bits en estos grupos, incluyendo signo, exponente y mantisa para FP32 y todos los bits para INT8.

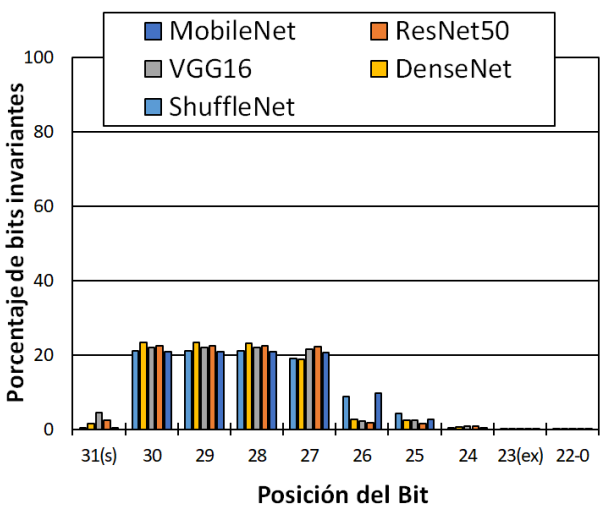


Fig. 2: Bits invariantes por posición (FP32)

En los modelos FP32 (Figura 1), la mayoría de los grupos muestran 4-6 bits redundantes, con ResNet50 teniendo hasta un 70 % de sus grupos con 4 bits redundantes. La Figura 2 indica que la redundancia se concentra en los bits 27-30, los más significativos del exponente, que son críticos para el valor del peso.

Para los modelos INT8, la redundancia es menor. En la Figura 3, MobileNet muestra que el 55 % de sus grupos no tienen redundancia de bits, mientras que otros modelos presentan una redundancia moderada de 1-3 bits. La Figura 4 muestra que los bits 4-6 son los más redundantes y deben ser protegidos, ya que tienen hasta un 80 % de redundancia y son cruciales para el valor del peso.

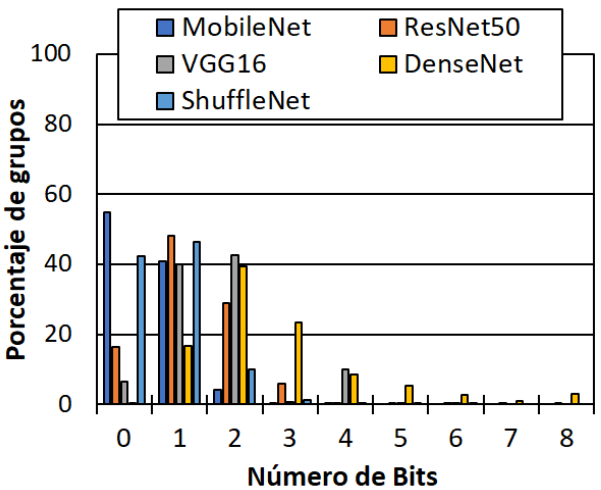


Fig. 3: Número de bits invariantes por grupos (INT8)

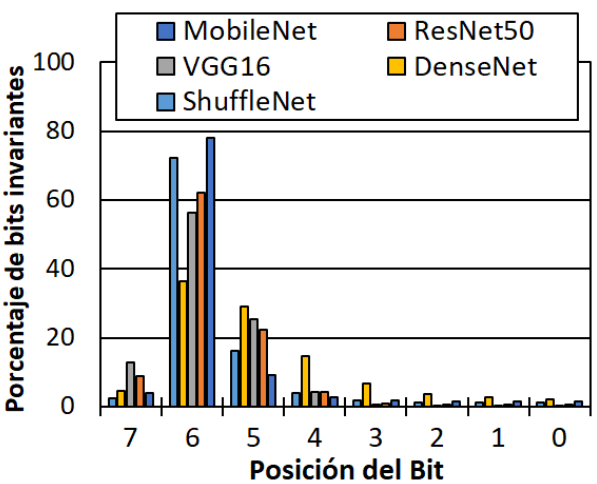


Fig. 4: Bits invariantes por posición (INT8)

IV. Resultados experimentales

Analizamos el impacto de los *bit flips* en el rendimiento de los modelos inyectando fallos aleatorios en los filtros convolucionales y evaluando la precisión con ImageNet[20]. Se utiliza el motor de inferencia Onnx Runtime-GPU (versión 1.12.0) [21] y la librería de python MxNet [22] en un equipo con procesador AMD EPYC 7282 16-Core y una tarjeta gráfica NVIDIA A100-40GB.

Se selecciona aleatoriamente una capa de convolución del modelo y se inyecta un *bit flip* aleatorio en el

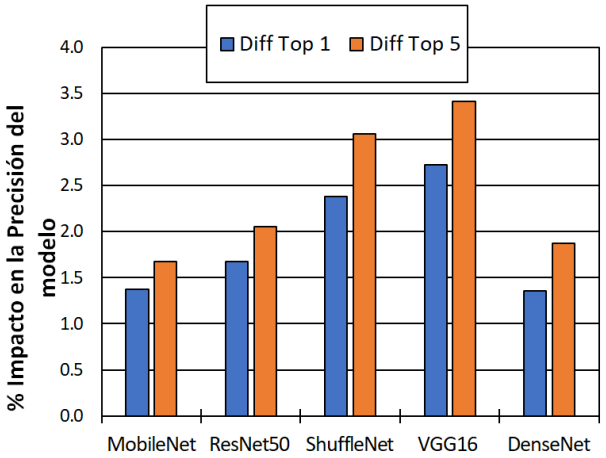


Fig. 5: Diferencia entre la precisión del modelo sin y con fallo con tipo de datos FP32 (mayor diferencia peor, dado que se pierde precisión)

tensor de pesos. Se evalúan tres métricas: precisión Top 1 y Top 5, sensibilidad del modelo a cambios de bits y porcentaje de fallos que producen un error en la salida (SDC).

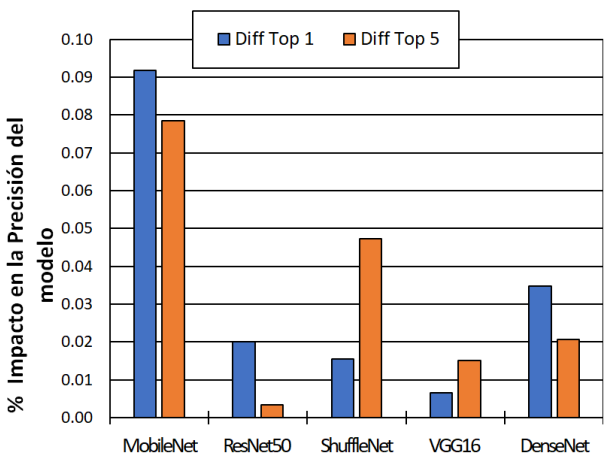


Fig. 6: Diferencia entre la precisión del modelo sin y con fallo con tipo de datos INT8 (mayor diferencia peor, dado que se pierde precisión)

Las Figuras 5 y 6 muestran el impacto de los *bit flips* en la precisión de los modelos. En FP32, la degradación varía del 1,35 % al 2,72 % en precisión Top 1 y del 1,67 % al 3,05 % en Top 5. En INT8, el impacto es menor, con un máximo de 0,092 % en Top 1 y 0,078 % en Top 5. Casos extremos en FP32 pueden llevar a una caída significativa en la precisión, mientras que en INT8 el impacto es mucho menor.

La segunda métrica mide las predicciones sensibles a fallos, observando que los modelos FP32 tienen un mayor porcentaje de predicciones sensibles y menos predicciones correctas en comparación con los modelos INT8.

La tercera métrica analiza el Ratio de SDCs, mostrando que los modelos FP32 no protegidos tienen un mayor porcentaje de imágenes clasificadas incorrectamente debido a un fallo, en comparación con los modelos INT8, que son menos críticos a los *bit flips* debido a su rango numérico limitado.

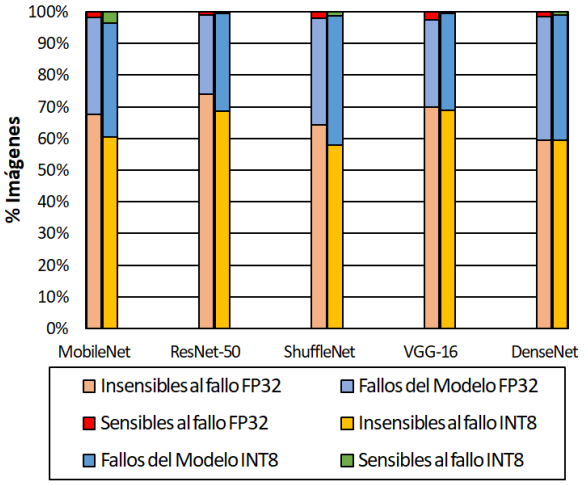


Fig. 7: Predicciones Imagen por Imagen para todo el *dataset* Imagenet

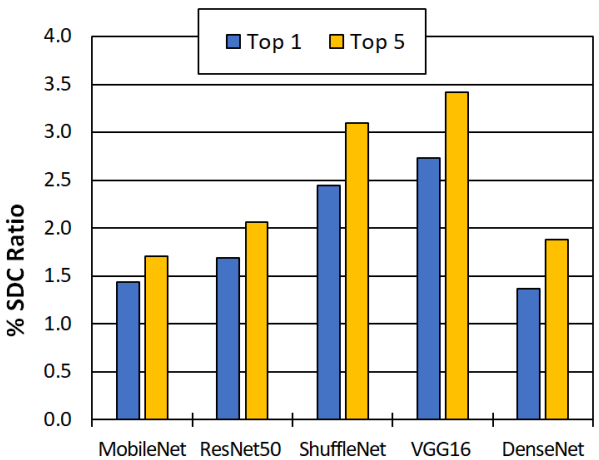


Fig. 8: Ratio de SDCs por modelo con tipo de dato FP32

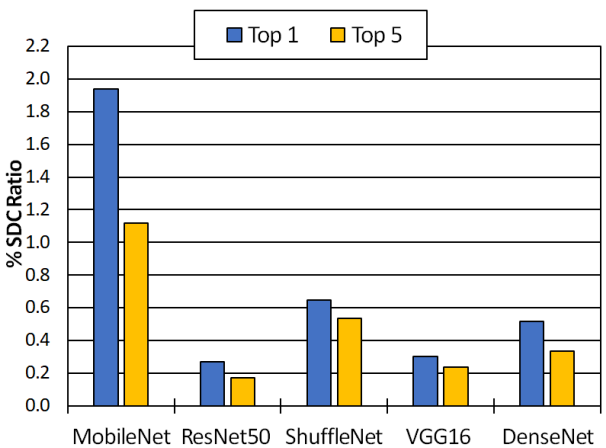


Fig. 9: Ratio de SDCs por modelo con tipo de Dato INT8

V. Conclusiones

En este artículo se muestra un amplio estudio dividido en dos partes. Primeramente, se ha llevado a cabo una búsqueda de bits invariantes dentro de los pesos en las operaciones de convolución según nuestra hipótesis inicial. Se ha demostrado por tanto que efectivamente, el hecho e que los pesos de la convolución estén en un mismo rango ya sea [-1,1] con coma flotante de 32 bits FP32 o [-128,128] en enteros hace

que los bits de mayor peso coincidan y por tanto sean un objetivo a proteger por nuestra parte en futuros trabajos.

En segundo lugar se ha realizando un análisis del impacto de bit flips en una variedad de modelos de redes neuronales convolucionales con números de coma flotante de 32 bits y enteros de 8 bits. Nuestros resultados muestran que, como era de esperar, los modelos de coma flotante son mucho más sensibles que los de enteros de 8 bits y que su punto crítico es el exponente. Proteger este rango de bits, como hemos demostrado, reduciría significativamente el impacto de los cambios de bit y sería razonable, dado que se podría aprovechar su mayor invariabilidad.

Agradecimientos

Este artículo ha sido realizado gracias a la dotación de la BECA FPU "PROGRAMA PROPIO DE LA UNIVERSITAT POLITÈCNICA DE VALÈNCIA – SUBPROGRAMA 1 (PAID-01-20)"No 20210037

Referencias

[1] Saad Albawi, Tareq Abed Mohammed, and Saad Al-Zawi, "Understanding of a convolutional neural network," in *2017 International Conference on Engineering and Technology (ICET)*, 2017, pp. 1–6, DOI: 10.1109/ICEngTechnol.2017.8308186.

[2] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," *CoRR*, vol. abs/1506.01497, 2015.

[3] Akhil Agnihotri, Prathamesh Saraf, and Kriti Rajesh Bapnad, "A convolutional neural network approach towards self-driving cars," *CoRR*, vol. abs/1909.03854, 2019.

[4] A. Vasuki and S. Govindaraju, *Deep neural networks for image classification*, pp. 27–49, 12 2017.

[5] Maria Bauza Nurullah Giray Kuru Tomás Lozano-Pérez Ferran Alet, Kenji Kawaguchi and Leslie Pack Kaelbling, "Tailoring: encoding inductive biases by optimizing unsupervised objectives at prediction time," 2018, `https://meta-learn.github.io/2020/papers/60_paper.pdf`.

[6] "How sparsity adds umph to ai inference," `https://blogs.nvidia.com/blog/2020/05/14/sparsity-ai-inference/`, 2021.

[7] Adnan Siraj Rakin, Zhezhi He, and Deliang Fan, "Bit-flip attack: Crushing neural network withprogressive bit search," *CoRR*, vol. abs/1903.12269, 2019.

[8] Sanghyun Hong, Pietro Frigo, Yigitcan Kaya, Cristiano Giuffrida, and Tudor Dumitras, "Terminal brain damage: Exposing the graceless degradation in deep neural networks under hardware fault attacks," *CoRR*, vol. abs/1906.01017, 2019.

[9] Daniel Alfonso Gonçalves Gonçalves de Oliveira, Laercio Lima Pilla, Thiago Santini, and Paolo Rech, "Evaluation and mitigation of radiation-induced soft errors in graphics processing units," *IEEE Transactions on Computers*, vol. 65, no. 3, pp. 791–804, 2016.

[10] Guanpeng Li, Siva Kumar Sastry Hari, Michael Sullivan, Timothy Tsai, Karthik Pattabiraman, Joel Emer, and Stephen W. Keckler, "Understanding error propagation in deep learning neural network (dnn) accelerators and applications," New York, NY, USA, 2017, Association for Computing Machinery.

[11] Le Ha Hoang, Muhammad Abdullah Hanif, and Muhammad Shafique, "Ft-clipact: Resilience analysis of deep neural networks and improving their fault tolerance using clipped activation," *CoRR*, vol. abs/1912.00941, 2019.

[12] Zitao Chen, Guanpeng Li, and Karthik Pattabiraman, "Ranger: Boosting error resilience of deep neural networks through range restriction," *CoRR*, vol. abs/2003.13874, 2020.

[13] Adam Neale and Manoj Sachdev, "Neutron radiation induced soft error rates for an adjacent-ecc protected sram in 28 nm cmos," *IEEE Transactions on Nuclear Science*, vol. 63, no. 3, pp. 1912–1917, 2016.

[14] Olivier Temam, "A defect-tolerant accelerator for emerging high-performance applications," vol. 40, no. 3, 2012.

[15] Zhezhi He, Adnan Siraj Rakin, Jingtao Li, Chaitali Chakrabarti, and Deliang Fan, "Defending and harnessing the bit-flip based adversarial weight attack," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 14083–14091.

[16] Yu Li, Min Li, Bo Luo, Ye Tian, and Qiang Xu, "Deepdyve: Dynamic verification for deep neural networks," *CoRR*, vol. abs/2009.09663, 2020.

[17] Jingtao Li, Adnan Siraj Rakin, Zhezhi He, Deliang Fan, and Chaitali Chakrabarti, "RADAR: run-time adversarial weight attack detection and accuracy recovery," *CoRR*, vol. abs/2101.08254, 2021.

[18] Florian Geissler, Syed Qutub, Sayanta Roychowdhury, Ali Asgari, Yang Peng, Akash Dhamasia, Ralf Graefe, Karthik Pattabiraman, and Michael Paulitsch, "Towards a safety case for hardware fault tolerance in convolutional neural networks using activation range supervision," *CoRR*, vol. abs/2108.07019, 2021.

[19] Onnx, "Model zoo github repository," 2023.

[20] ImageNet, "Data set," 2023.

[21] Onnx Runtime, "Cross-platform inference and training machine-learning accelerator," 2023.

[22] Apache MxNet, "Library for deep learning," 2023.

[23] Rick Salay, Rodrigo Queiroz, and Krzysztof Czarnecki, "An analysis of ISO 26262: Using machine learning safely in automotive software," *CoRR*, vol. abs/1709.02435, 2017.

[24] Wenshuo Li, Guangjun Ge, Kaiyuan Guo, Xiaoming Chen, Qi Wei, Zhen Gao, Yu Wang, and Huazhong Yang, "Soft error mitigation for deep convolution neural network on fpga accelerators," in *2020 2nd IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, 2020, pp. 1–5.

[25] Muhammad Hanif and Muhammad Shafique, "Salvagednn: salvaging deep neural network accelerators with permanent faults through saliency-driven fault-aware mapping," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 378, pp. 20190164, 02 2020.

[26] Jae-San Kim and Joon-Sung Yang, "Dris-3: Deep neural network reliability improvement scheme in 3d die-stacked memory based on fault analysis," 06 2019, pp. 1–6.

[27] Guanpeng Li, Siva Kumar Sastry Hari, Michael Sullivan, Timothy Tsai, Karthik Pattabiraman, Joel Emer, and Stephen W. Keckler, "Understanding error propagation in deep learning neural network (dnn) accelerators and applications," New York, NY, USA, 2017, Association for Computing Machinery.

[28] Jingtao Li, Adnan Siraj Rakin, Yan Xiong, Liangliang Chang, Zhezhi He, Deliang Fan, and Chaitali Chakrabarti, "Defending bit-flip attack through dnn weight reconstruction," in *2020 57th ACM/IEEE Design Automation Conference (DAC)*, 2020, pp. 1–6.

[29] Mojan Javaheripi and Farinaz Koushanfar, "HASHTAG: hash signatures for online detection of fault-injection attacks on deep neural networks," *CoRR*, vol. abs/2111.01932, 2021.

[30] Syed Qutub, Florian Geissler, Yang Peng, Ralf Gräfe, Michael Paulitsch, Gereon Hinz, and Alois Knoll, "Hardware faults that matter: Understanding and estimating the safety impact of hardware faults on object detection dnns," Berlin, Heidelberg, 2022, Springer-Verlag.

[31] Brandon Reagen, Udit Gupta, Lillian Pentecost, Paul Whatmough, Sae Kyu Lee, Niamh Mulholland, David Brooks, and Gu-Yeon Wei, "Ares: A framework for quantifying the resilience of deep neural networks," New York, NY, USA, 2018, Association for Computing Machinery.