Redes Convolucionales aplicado a un Rover

1er Arbués Enrique Pérez Villegas
Departamento de Ciencias de la Computación, Universidad Nacional de Ingenieria, Lima, Perú
arbues.perez.v@uni.pe

***Resumen*—Este documento explora la aplicación de redes neuronales convolucionales (CNN) en la robótica, con énfasis en el uso de plataformas como Raspberry Pi 5 y Jetson Nano para tareas de visión por computadora.**

***Index Terms*—Redes neuronales convolucionales, visión por computadora, robótica, Raspberry Pi, Jetson Nano**

I. Introducción

Las redes neuronales convolucionales (CNN) han emergido como una de las herramientas más poderosas en el campo de la inteligencia artificial, especialmente en aplicaciones de visión por computadora. En el contexto de la robótica, estas redes han permitido avances significativos en la capacidad de los robots para percibir y comprender su entorno, una habilidad crítica en competencias como el European Rover Challenge (ERC).

En estas competencias, los robots están diseñados para ejecutar una variedad de tareas complejas que simulan escenarios de exploración en otros planetas, como Marte. Estas tareas incluyen la navegación autónoma en terrenos irregulares, la detección y clasificación de objetos, y la manipulación de herramientas para la recolección de muestras. La implementación de CNN en estos robots permite mejorar la precisión y la eficiencia en la interpretación de las imágenes capturadas por las cámaras a bordo, lo cual es esencial para la toma de decisiones en tiempo real.

El uso de plataformas de hardware como la Raspberry Pi 5 y Jetson Nano proporciona un entorno flexible y potente para desplegar modelos de CNN. Estas plataformas, aunque limitadas en capacidad computacional en comparación con servidores dedicados, son suficientes para ejecutar modelos optimizados que pueden cumplir con los requisitos de las competencias robóticas. Además, su bajo consumo de energía y su capacidad para integrarse con otros componentes del robot, como sensores y actuadores, las hacen ideales para escenarios de competencia donde la autonomía y la eficiencia energética son cruciales.

Este documento explora las metodologías y consideraciones clave para implementar CNN en robots que participan en competencias como el ERC, destacando los pasos necesarios para seleccionar el hardware adecuado, configurar el entorno de desarrollo, y diseñar e integrar los modelos de CNN en el sistema robótico. Al proporcionar una guía detallada, esperamos ayudar a los equipos a maximizar el rendimiento de sus robots en escenarios de competencia extrema.

II. Fundamentos de Redes Neuronales Convolucionales

Las redes neuronales convolucionales (CNN) son una clase de redes neuronales profundas diseñadas específicamente para procesar datos con una estructura de cuadrícula, como las imágenes. Son ampliamente utilizadas en aplicaciones de visión por computadora debido a su capacidad para capturar características espaciales y patrones jerárquicos en los datos [1].

II-A. Arquitectura General de las CNN

La arquitectura general de una CNN se compone de varias capas que procesan los datos de entrada de manera secuencial. Las principales capas incluyen capas convolucionales, capas de pooling, capas de activación y capas completamente conectadas. Estas capas trabajan juntas para extraer características, reducir la dimensionalidad, y finalmente, clasificar las entradas. Una CNN típica puede estar compuesta por decenas o incluso cientos de estas capas, dependiendo de la complejidad de la tarea a realizar [6].

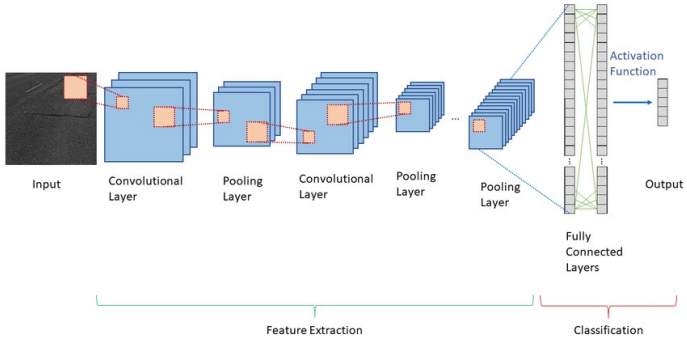

Figura 1: Arquitectura general de una red neuronal convolucional (CNN) [6].

II-B. Capas Convolucionales

Las capas convolucionales son el núcleo de las CNN. En estas capas, se aplican filtros (también conocidos como kernels) sobre la entrada para extraer características locales como bordes, texturas y patrones. Estos filtros se desplazan sobre la imagen de entrada y realizan una operación de convolución para generar un mapa de características. La profundidad del filtro determina la cantidad de características que se extraen [3].

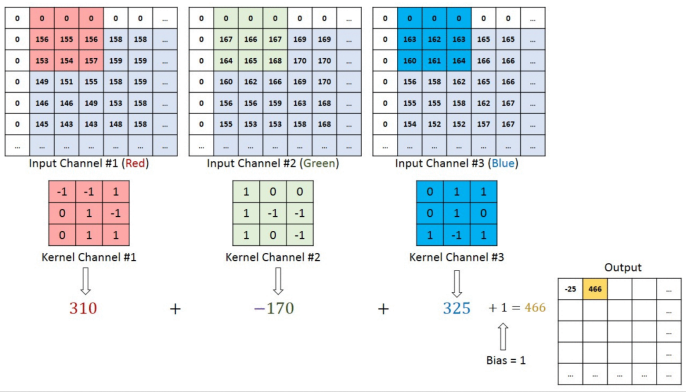

Figura 2: Operación de convolución aplicada a una imagen de entrada [3].

II-C. Funciones de Activación

Las funciones de activación introducen no linealidades en la red, permitiendo que la CNN capture relaciones más complejas en los datos. Una de las funciones de activación más comunes es la ReLU (Rectified Linear Unit), definida como:

$$f(x) = \max(0, x)$$

Otra función de activación común es la Sigmoide, dada por:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

Estas funciones transforman la salida de las capas convolucionales y aseguran que la red pueda modelar relaciones no lineales [4].

II-D. Pooling

La operación de pooling se utiliza para reducir la dimensionalidad espacial de las características extraídas y, al mismo tiempo, hacer que la red sea más robusta a variaciones en la posición de las características. Las dos técnicas de pooling más comunes son el max-pooling y el average-pooling. El max-pooling selecciona el valor máximo en una ventana específica, mientras que el average-pooling calcula el promedio de los valores en la ventana [5].

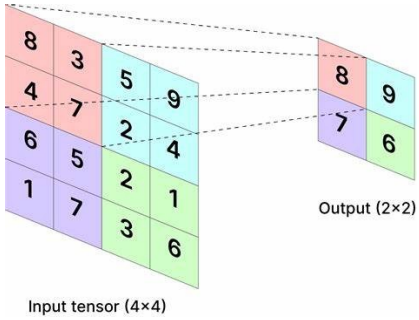

Figura 3: Ejemplo de max-pooling aplicado a un mapa de características [5].

II-E. Capas Fully Connected

Las capas completamente conectadas (fully connected) se utilizan en las etapas finales de una CNN para realizar la clasificación. Cada neurona en estas capas está conectada a todas las neuronas de la capa anterior. Estas capas toman las características extraídas por las capas convolucionales y de pooling y las utilizan para tomar decisiones finales, como asignar una clase a la imagen de entrada [3].

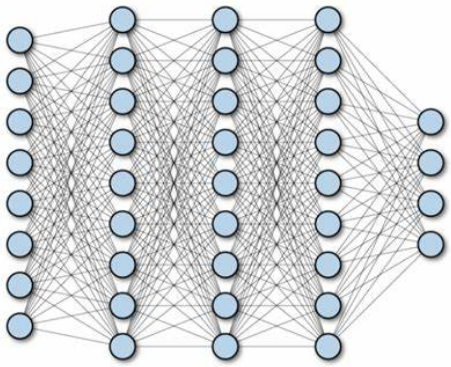

Figura 4: Ejemplo de una capa fully connected en una CNN [3].

III. Aplicaciones de CNN en Visión por Computadora

Las redes neuronales convolucionales (CNN) son ampliamente utilizadas en diversas aplicaciones de visión por computadora. A continuación, se presentan algunas de las aplicaciones más comunes y su relevancia en la robótica.

III-A. Clasificación

La clasificación de imágenes es una de las aplicaciones más fundamentales de las CNN. En esta tarea, la red neuronal asigna una etiqueta a la imagen completa, identificando el objeto principal que contiene. Esta aplicación es esencial para robots que necesitan identificar objetos antes de interactuar con ellos.

Figura 5: Ejemplo de clasificación con AlexNet, VGGNet, y ResNe.

III-B. *Segmentación Semántica*

La segmentación semántica consiste en clasificar cada píxel de la imagen en una categoría específica, lo que permite identificar y diferenciar entre diferentes objetos o regiones en una imagen. En robótica, esta técnica es útil para mapear el entorno y tomar decisiones basadas en la comprensión detallada del espacio.

Figura 6: Ejemplo de segmentación semántica con FCN, U-Net, y DeepLab.

III-C. *Detección de Objetos*

La detección de objetos extiende la clasificación al identificar no solo la presencia de un objeto en una imagen, sino también su ubicación exacta mediante cuadros delimitadores (bounding boxes). Esta aplicación es crucial para robots que necesitan identificar y localizar objetos específicos en su entorno para interactuar con ellos.

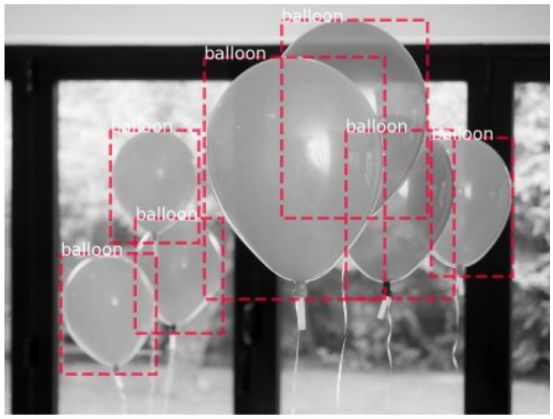

Figura 7: Ejemplo de detección de objetos con Faster R-CNN, YOLO, y SSD.

III-D. *Segmentación de Instancias*

La segmentación de instancias combina la segmentación semántica y la detección de objetos para identificar cada instancia individual de un objeto en la imagen, incluso si pertenecen a la misma clase. Esto es especialmente útil en escenarios donde es necesario diferenciar entre múltiples objetos similares, como en tareas de manipulación robótica.

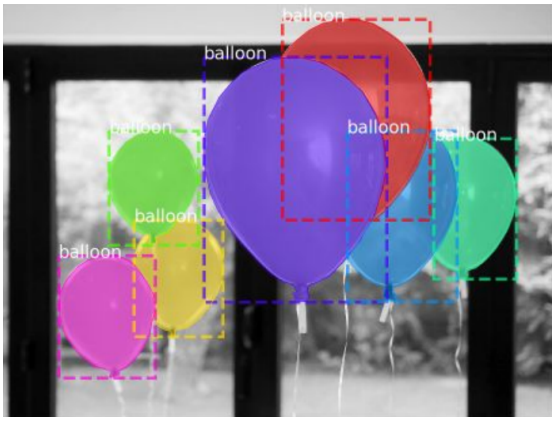

Figura 8: Ejemplo de segmentación de instancias con Mask R-CNN, PANet, y YOLACT.

IV. Implementación General de CNN en Robótica

La implementación de redes neuronales convolucionales (CNN) en robótica requiere una planificación cuidadosa, especialmente en cuanto a la selección de hardware, la configuración del entorno de desarrollo, el diseño del modelo y la integración con otros componentes del robot. A continuación se detalla un enfoque general para llevar a cabo esta implementación utilizando plataformas populares como Raspberry Pi 5 y Jetson Nano.

IV-A. *Seleccionando la Plataforma de Hardware*

La elección entre Raspberry Pi 5 y Jetson Nano depende de las necesidades específicas del proyecto y las tareas de visión por computadora que el robot debe realizar.

- **Raspberry Pi 5**: Ideal para aplicaciones donde el bajo consumo de energía y el costo son primordiales. Es adecuado para modelos de CNN ligeros que no requieren un gran poder de procesamiento.
- **Jetson Nano**: Ofrece mayor capacidad de procesamiento gráfico gracias a su GPU integrada, lo que lo hace más adecuado para aplicaciones que requieren la ejecución de modelos más complejos o en tiempo real.

Cuadro I: Comparación de características entre Raspberry Pi 5 y Jetson Nano

Característica	**Raspberry Pi 5**	**Jetson Nano**
CPU	Quad-core Cortex-A72	Quad-core ARM A57
GPU	VideoCore VI	128-core Maxwell GPU
RAM	4GB LPDDR4	4GB LPDDR4
Consumo de Energía	Bajo	Moderado
Costo	Bajo	Moderado

IV-B. *Configuración del Entorno de Desarrollo*

La configuración del entorno de desarrollo es crucial para garantizar que el hardware seleccionado pueda ejecutar eficientemente los modelos de CNN.

- **Instalación de sistemas operativos y dependencias**: Para ambas plataformas, se recomienda utilizar sistemas operativos ligeros como Raspberry Pi OS para Raspberry

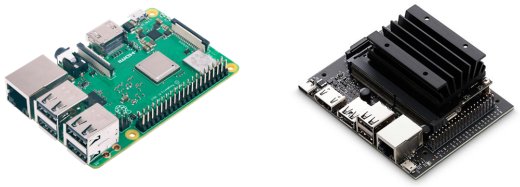

Figura 9: Comparación física y de especificaciones entre Raspberry Pi 5 y Jetson Nano.

Pi 5 y JetPack para Jetson Nano. Es necesario instalar las dependencias básicas como Python, pip, y bibliotecas específicas de IA.

- **Configuración de herramientas y bibliotecas necesarias**: Tanto TensorFlow como PyTorch son opciones populares para desarrollar y entrenar modelos de CNN. En Jetson Nano, se puede aprovechar la GPU con TensorFlow-GPU o PyTorch con CUDA para acelerar la inferencia.

IV-C. Desarrollo del Prototipo de CNN

El desarrollo de un prototipo de CNN adaptado a las limitaciones de hardware es un paso clave en la implementación.

- **Diseño de la arquitectura de CNN adaptada a las limitaciones de hardware**: Considerando las limitaciones de RAM y procesamiento, es importante diseñar una CNN ligera o utilizar una ya existente como MobileNet o SqueezeNet, que están optimizadas para dispositivos de baja potencia.
- **Implementación de un modelo básico de CNN para reconocimiento de imágenes**: Se puede comenzar con un modelo preentrenado, como MobileNet, y realizar fine-tuning para adaptarlo a las necesidades específicas del robot. Esto reduce el tiempo de desarrollo y mejora la precisión en tareas específicas como la detección de objetos o la clasificación de imágenes.

Cuadro II: Modelos de CNN optimizados para dispositivos de bajo consumo

Modelo	Tamaño del Modelo	Precisión
MobileNetV2	14 MB	71.8 %
SqueezeNet	5 MB	57.5 %
EfficientNet-Lite	20 MB	78.3 %

IV-D. Integración con Sensores y Actuadores

La integración de la CNN con los sensores y actuadores del robot es esencial para permitir que el robot interactúe de manera efectiva con su entorno.

- **Conexión de cámaras y otros sensores al robot**: La cámara es el sensor principal para la visión por computadora. Es crucial configurar la cámara para que funcione correctamente con la plataforma seleccionada y optimizar la captura de imágenes para la inferencia de la CNN.
- **Interacción de la CNN con otros componentes del robot para realizar tareas de navegación y detección**: La salida de la CNN puede ser utilizada para tomar decisiones en tiempo real, como evitar obstáculos, seguir líneas, o identificar y recoger objetos. Esta integración debe ser cuidadosamente coordinada para garantizar una respuesta rápida y precisa del robot.

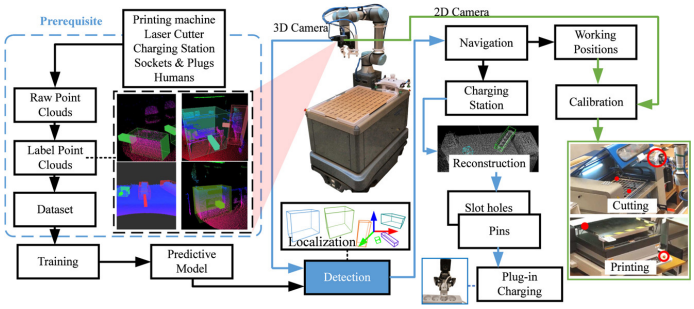

Figura 10: Esquema de integración de la CNN con sensores y actuadores en un robot.

V. Desafíos y Consideraciones

La implementación de redes neuronales convolucionales (CNN) en robótica, especialmente en entornos extremos como Marte, presenta una serie de desafíos y consideraciones que deben ser abordados para garantizar el éxito de la misión.

V-A. Limitaciones Computacionales

Los robots que operan en Marte deben utilizar hardware de bajo consumo y limitado en capacidad de procesamiento. Esto significa que los modelos de CNN deben ser ligeros en términos de cantidad de parámetros para poder ejecutarse eficientemente en plataformas como Raspberry Pi o Jetson Nano. La selección y optimización de modelos es crucial para mantener un balance entre precisión y velocidad de inferencia.

V-B. Procesamiento en Tiempo Real

En aplicaciones robóticas, la capacidad de procesar datos en tiempo real es fundamental. La CNN debe ser capaz de analizar imágenes y tomar decisiones rápidas para permitir que el robot navegue y opere de manera autónoma. Esto requiere un equilibrio cuidadoso entre la complejidad del modelo y la latencia aceptable para el sistema.

V-C. Impacto del Entorno en la Captura de Imágenes

El entorno marciano presenta desafíos únicos para la captura de imágenes, incluyendo la baja iluminación, la presencia de polvo en la atmósfera, y las variaciones extremas de temperatura. Estas condiciones pueden afectar la calidad de las imágenes capturadas, lo que a su vez impacta la precisión de la CNN en la detección y clasificación de objetos. Es necesario considerar el uso de técnicas de preprocesamiento de imágenes para mitigar estos efectos.

V-D. Manejo de la Resolución y Tipo de Imágenes

La resolución y el tipo de imágenes que el robot capturará tienen un impacto directo en el rendimiento de la CNN. Imágenes de alta resolución proporcionan más detalles, pero requieren más poder de procesamiento y memoria. Por otro lado, imágenes de baja resolución pueden acelerar el procesamiento, pero a costa de perder precisión. Es crucial encontrar un equilibrio adecuado según las capacidades del hardware disponible y las necesidades de la misión.

V-E. Robustez frente a las Condiciones Climáticas Extremas

El frío extremo en Marte puede afectar tanto al hardware como a los algoritmos de visión por computadora. Los componentes electrónicos deben estar diseñados para operar en temperaturas bajas, y los algoritmos deben ser lo suficientemente robustos para funcionar bajo estas condiciones adversas sin pérdida significativa de precisión.

VI. Conclusiones y Futuras Direcciones

En este documento se ha explorado la implementación de redes neuronales convolucionales (CNN) en robótica, con un enfoque específico en plataformas como Raspberry Pi 5 y Jetson Nano. Se han discutido las aplicaciones de las CNN en tareas de visión por computadora, así como los desafíos que presentan los entornos extremos como Marte.

El futuro de la implementación de CNN en robótica marciana presenta varias direcciones prometedoras. En primer lugar, el desarrollo de modelos más ligeros y eficientes, como MobileNet y EfficientNet, permitirá ejecutar tareas complejas de visión por computadora, como detección de objetos, segmentación semántica, segmentación de instancias y mapeo de zonas, con menor consumo de recursos. Estos modelos, junto con técnicas de compresión de redes y optimización de inferencia, serán clave para superar las limitaciones de hardware.

Además, el avance en técnicas de fusión sensorial permitirá a los robots combinar datos visuales con información de otros sensores, mejorando la precisión y la robustez en la toma de decisiones. La integración de aprendizaje por refuerzo y métodos de adaptación en línea también puede proporcionar a los robots una mayor autonomía y capacidad de aprendizaje en tiempo real, ajustándose mejor a las condiciones cambiantes del entorno marciano.

En conclusión, la implementación de CNN en robótica ofrece un gran potencial para la exploración planetaria, pero requiere una planificación cuidadosa y un enfoque multidisciplinario para abordar los desafíos inherentes a estos entornos extremos. El desarrollo continuo de modelos especializados y la mejora de las capacidades de hardware serán fundamentales para el éxito de futuras misiones robóticas en Marte y más allá.

Referencias

[1] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition,"*Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278-2324, 1998.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, Ïmagenet classification with deep convolutional neural networks,"*Advances in Neural Information Processing Systems*, vol. 25, pp. 1097-1105, 2012.

[3] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning,"*Nature*, vol. 521, no. 7553, pp. 436-444, 2015.

[4] V. Nair and G. E. Hinton, Rectified linear units improve restricted boltzmann machines,ïn *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pp. 807-814, 2010.

[5] D. Scherer, A. Müller, and S. Behnke, .Evaluation of pooling operations in convolutional architectures for object recognition,ïn *International Conference on Artificial Neural Networks*, pp. 92-101, 2010.

[6] A. Krizhevsky, I. Sutskever, and G. E. Hinton, Ïmagenet classification with deep convolutional neural networks,"*Advances in Neural Information Processing Systems*, vol. 25, pp. 1097-1105, 2012.

[7] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation,ïn *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3431-3440, 2015.

[8] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks,"*Advances in Neural Information Processing Systems*, vol. 28, pp. 91-99, 2015.

[9] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN,ïn *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2961-2969, 2017.

[10] NVIDIA Corporation, "Getting Started with Jetson Nano Developer Kit,"NVIDIA Developer Documentation, 2021. [Online]. Available: https://developer.nvidia.com/embedded/jetson-nano-developer-kit