Clasificación de daños con visión artificial en transporte marítimo de vehículos: Implementación de Redes Neuronales Convolucionales

Hugo Daniel Flores [1], Carlos Neil [2]

Universidad Abierta Interamericana. Facultad de Tecnología Informática.
Centro de Altos Estudios en Tecnología Informática. Buenos Aires. Argentina.
[1] hugodaniel.flores@alumnos.uai.edu.ar
[2] Carlos.Neil@uai.edu.ar

Resumen. Un alto porcentaje de vehículos son transportados vía marítima, donde son manipulados según protocolos específicos. Controlar su estado y detectar daños recae en personal capacitado, lo que genera dificultades al momento de imputar responsabilidades. Para mejorar la eficiencia de estos procesos, la Visión Artificial (VA) se ha consolidado como una alternativa viable para detectar y clasificar daños en distintas industrias. El objetivo de este trabajo es implementar el uso de algoritmos de Inteligencia Artificial (IA) para detectar y clasificar daños en la industria marítima. Se describe la creación y ejecución de 3 modelos de Redes Neuronales Convolucionales (RNC) para identificar tipos de daños, partes de autos dañados, y lugares donde son producidos, así como el procedimiento para obtener la información de las bases de datos actualmente en producción en la industria automotriz. Luego, se realiza un análisis profundo para determinar la exactitud de los resultados obtenidos y los cambios que se deben hacer en los procesos para producir mejores modelos de RNC. Se ha comprobado que las técnicas de VA basadas en RNC son altamente efectivas para implementar soluciones destinadas a la detección y clasificación de daños en esta industria. También se ha demostrado que es conveniente para la industria marítima estandarizar tanto los procesos de inspección como los procedimientos de captura de datos para producir modelos de IA con mejores resultados.

Palabras clave: Daños, Redes Neuronales Convolucionales, Visión Artificial

1 Introducción

1.1 Antecedentes

La producción de daños en vehículos es un tema recurrente en la industria marítima y en particular en la industria automotriz. Los sistemas actuales tratan la información con los lineamientos convencionales como la aplicación de códigos alfanuméricos para especificar las partes, los tipos de avería y la dimensión de ellas; y desde hace algunos años a esos códigos se han sumado las capturas de imágenes para certificar visualmente los daños. Como resultado de toda la información colectada se generan reportes

electrónicos que tienen como destino a todas las empresas que intervienen en un punto de cambio de responsabilidad con el fin de definir la imputabilidad de los daños, y con ello determinar a quienes corresponden los costos de reparación. Los problemas se presentan cada vez que se detecta un daño y este debe ser imputado a una empresa, lo cual muchas veces no es claro porque el daño puede haber sido provocado no por esa empresa sino por otra en un punto de tránsito previo (H. Flores et al., 2009). En consecuencia, la identificación de los daños les brinda a las empresas información para tomar medidas preventivas para evitar que los daños sucedan nuevamente, y además también les permite definir una estructura de costos para las reparaciones. También existen problemas relacionados con los daños ocurridos en lugares donde la vista humana no llega, como pueden ser lugares bajos, o debido a la rapidez con la que hay que realizar una inspección se dejan pasar por alto observaciones, o en casos donde la luz o las condiciones del tiempo no son favorables para la visión humana (Larcher et al., 2013). Es importante remarcar que los daños buscados son de tamaño pequeño y visibilidad casi nula en función de la distancia en algunos casos.

Es decir, los lugares donde se presentan los problemas son en los cambios de responsabilidad: en la subida a buque, la bajada de buque o dentro del buque; y además es probable también encontrar daños en el ingreso o la salida de puerto, y también pueden producirse daños inclusive en la estiba en puerto. La importancia del tema en cuestión radica en que la evaluación actual de daños depende de la visión humana y del criterio y experiencia del perito, y en muchos casos los errores o fallas resultan sumamente costosos (Russo et al., 2022). También, en la actualidad se presentan inconvenientes por las demoras en la emisión de informes en tiempo y forma. Toda información relacionada con un daño debe ser claramente documentada para las partes interesadas en el producto: fabricantes, seguros, transportes, puertos y operadores logísticos. Demoras en los procesos administrativos produce demoras en la entrega de productos y pérdidas a las empresas (Santome Jiménez, 2020).

1.2 Procesos

La utilidad de la solución propuesta producirá mejoras en todo el proceso de control de daños tanto directamente en lugar de inspección, como así también en la búsqueda de información en las bases de datos a través de las interfaces correspondientes. En la actualidad, toda información capturada es almacenada en bases de datos por lo que se cuenta con grandes bancos de imágenes (H. D. Flores et al., 2023). Este artículo propone desarrollar un clasificador de imágenes creado mediante técnicas de aprendizaje supervisado con RNC, una de las arquitecturas de aprendizaje profundo más populares, diseñadas específicamente para resolver problemas de VA, como el reconocimiento de patrones y la clasificación de imágenes. En la actualidad, las inspecciones se desarrollan con celulares y aplicaciones instaladas a medida, y estas permiten codificar los daños encontrados y tomar fotografías. Hoy en día no se cuenta con información acerca de una herramienta inteligente en la industria marítima que permita a los inspectores resolver la problemática de clasificación de daños (Flórez et al., 2022).

La presentación de este artículo tiene 3 objetivos muy importantes para la industria:

1) Las RNC son aplicables a este tipo de procesos de inspección donde es complejo identificar daños y/o descubrir faltantes. 2) La aplicación de VA para demostrar a las empresas involucradas en todo el proceso logístico que la IA resuelve este tipo de problemáticas. 3) Mostrar a las empresas, cada una con sus propios estándares de inspección, que la normalización de los procesos es necesaria para mejorar los modelos.

2 Materiales y Métodos

2.1 Redes Neuronales Convolucionales

Las RNC son un método de aprendizaje automático que permite a las máquinas reconocer patrones y hacer predicciones. Son un tipo de Red Neuronal (RN) que se especializa en procesar datos espaciales, como imágenes, y es por ello qué se utilizan para las problemáticas como las de este artículo (Comprender Las Redes Neuronales Convolucionales (CNN), 2025). Las RN identifican patrones simples o clasifican información básica, y las RNC solucionan problemas de VA que son más complejos. En este trabajo la complejidad radica en que los daños pueden ser de tamaño muy pequeño y de difícil identificación, pero la imputación de ellos y la distribución de costos son de significativa importancia para toda la cadena logística. Las RNC procesan capas imitando al córtex visual del cerebro humano para identificar distintas características en las entradas que hacen que se pueda identificar formas. Su principal ventaja es que cada parte de la red es entrenada para realizar una tarea específica, aprendiendo diferentes niveles de abstracción, por lo que se reduce significativamente el número de conexiones en las capas ocultas y el entrenamiento es más rápido (Salina et al., 2024). Las RNC son muy potentes debido a que contienen varias capas ocultas especializadas y con una jerarquía, esto significa que las primeras capas pueden detectar características básicas como líneas, curvas o bordes y se van especializando hasta llegar a capas más profundas capaces de reconocer formas complejas como objetos o daños (Practical Deep Learning for Coders: YouTube, 2025). Los tres componentes principales de una RNC son:

1. Capas de convolución: constituyen el núcleo y su función principal es extraer características de las imágenes de entrada. Funciones: A) Filtrado convolucional: las capas de convolución aplican filtros (kernel) a la imagen de entrada. Un filtro es una pequeña matriz de tamaño 3x3 o 5x5, que pasa (o "convoluciona") sobre la imagen. B) Detección de características: cada filtro detecta tipos de características, como bordes, texturas o patrones específicos. C) Mapeo de características: el resultado de aplicar un filtro a la imagen es un mapa de características. Cada capa convolucional produce mapas de características correspondientes a cada filtro. D) No linealidad: tras aplicar el filtro, se aplica una función de activación no lineal, como ReLU (Rectified Linear Unit), para introducir no linealidad al modelo. Esto permite captar relaciones más complejas en los datos.
2. Capas de agrupamiento: o submuestreo o subredes. La capa de agrupamiento reduce la dimensionalidad de los datos reteniendo sólo las características más importantes. Esto reduce el número de parámetros y el riesgo de sobreaprendizaje. La agrupación reduce el tamaño de los mapas de características, lo que a su vez reduce el número de parámetros y cálculos necesarios en la red. Esto hace que el modelo

sea más eficiente. Tipos: A) Max-Pooling: es el método más común. Divide la imagen en subregiones no superpuestas y toma el valor máximo de cada subregión. En una región de 2x2, Max-Pooling tomará el valor más alto de los cuatro píxeles. B) Average-Pooling: es otro método en el que se promedian los valores de cada subregión. Es menos agresivo que el Max-Pooling, pero conserva menos detalles.

3. Capas totalmente conectadas: suelen encontrarse al final de una RNC y actúan como clasificadores de las características extraídas por las capas anteriores. Estas capas se utilizan para el razonamiento de alto nivel, explotando funciones de activación como Softmax para la clasificación. Softmax sirve para clasificar adecuadamente las entradas, produciendo una probabilidad de 0 a 1. Tipos: A) Conexión total: en estas capas, cada neurona está conectada a todas las neuronas de la capa anterior. Esto permite combinar las características extraídas para formar una representación global de la imagen. B) Clasificación: las capas totalmente conectadas toman las características aprendidas y las transforman en salidas. La salida podría ser un vector de probabilidades que representa las clases posibles. C) Función de activación: estas neuronas suelen utilizar funciones de activación Softmax para problemas de clasificación multiclase. Existe también la función Sigmoide que se utiliza para clasificación binaria. D) Pesos de aprendizaje: durante el entrenamiento, los pesos de estas conexiones se ajustan para minimizar el error de predicción. Las capas totalmente conectadas desempeñan un papel clave en la generalización del modelo para datos no vistos (Comprender Las Redes Neuronales Convolucionales (CNN), 2025).

Por otro lado, el entrenamiento de una RNC se basa en la retropropagación, un proceso iterativo que ajusta los pesos de la red para minimizar una función de pérdida que describe la desviación entre las predicciones del modelo y los valores reales de los datos de entrenamiento. Cuando se entrena una RNC, se utilizan técnicas de optimización y regularización para mejorar la eficiencia del aprendizaje y mitigar el sobreaprendizaje. Una función de clasificación y optimización normalmente utilizada es Adam. La tasa de aprendizaje (lr) establece el control de cuánto se están ajustando los pesos en una red con respecto al gradiente de pérdida. Cuanto menor sea, más lento el entrenamiento (Uso de PyTorch Para Entrenar El Modelo de Clasificación de Imágenes | Microsoft Learn, 2025).

2.2 Base de datos

Los datos para este proyecto son extraídos de sistemas web en producción (7x24) de una empresa que es contratada para controlar la importación y exportación de autos 0 KM de Argentina (Austral Marine Services, 2025). Los datos corresponden a los últimos 10 años de inspecciones en puerto en la descarga y carga a un buque, y en el ingreso y salida de puerto o estiba. Los tamaños de imágenes con los que se hicieron los entrenamientos varían de 32x32 a 256x256 píxeles. Antes de introducir las imágenes en los modelos y someterlas a las tres capas mencionadas, las imágenes son preprocesadas para garantizar que los datos estén en un formato óptimo para el aprendizaje. Se aplicaron estos pasos típicos del preprocesamiento de imágenes:

1. Cambio de tamaño: las imágenes son de distintos tamaños, pero las RNC suelen exigir que todas las imágenes de entrada tengan el mismo tamaño. En consecuencia, cada imagen se redimensiona a un único tamaño.
2. Normalización: consiste en ajustar los valores de los píxeles para que se sitúen dentro de un rango común, entre 0 y 1 o -1 y 1. Esto acelera la convergencia durante el entrenamiento y mejora la estabilidad del modelo.
3. Centrado y calibrado: algunas aplicaciones podrían necesitar centrar los datos en torno a cero restando la media de los valores de los píxeles y dividiendo por la desviación típica.
4. Aumento de los datos: técnica que consiste en aplicar transformaciones aleatorias a la imagen de entrenamiento para crear variaciones. Esto favorece a que el modelo sea más robusto al enseñarle a reconocer formas a pesar de las posibles variaciones. Algunas técnicas son: rotación, zoom, flip, modificación del brillo y el contraste.

Por lo tanto, el preprocesamiento de imágenes es un paso importante especialmente en este tipo de sistemas, ya que garantiza que todas las imágenes tengan igual tamaño y formato, lo que facilita el aprendizaje del modelo. Aquí se normalizaron los datos entre 0 y 1, y –1 y 1 para estabilizar los entrenamientos y acelerar la convergencia. Aumentar los datos permitió que el modelo generalice mejor al aprender de variaciones más amplias en los datos de entrenamiento. También ha sido necesario en esta etapa eliminar automáticamente imágenes fuera de contexto (las que contienen información de identificación de los vehículos e imágenes involucradas en más de un tipo de daño).

2.3 Herramientas de Software

A continuación, se describen las herramientas utilizadas. Todas estas son librerías y frameworks de uso estándar en tecnología, y desarrollados por distintas empresas.

Anaconda: es una distribución de software libre y de código abierto de los lenguajes Python y R utilizada en ciencia de datos y Machine Learning (ML). (Advance AI with Open Source | Anaconda, 2025).

Google Colab: es un servicio cloud, basado en los Notebooks de Jupyter, que permite el uso gratuito de las GPU y TPU de Google (Colab.Google, 2025).

Jupyter Notebook: es un entorno informático interactivo basado en la web para crear documentos Jupyter. Un documento Jupyter es un documento JSON, que sigue un esquema versionado y que contiene una lista ordenada de celdas de entrada y salida (Project Jupyter | Home, 2025).

Pytorch (PT): es una biblioteca de aprendizaje automático de código abierto basada en la biblioteca de Torch, utilizado para aplicaciones de VA y procesamiento de lenguajes naturales. Es un software libre. PT proporciona 2 características de alto nivel: 1) computación de tensores con una aceleración fuerte en GPU; 2) RN profundas construidas en un sistema de diferenciación automática de bases de datos (PyTorch, 2025).

ONNX: es un formato abierto para modelos de ML, que permite intercambiar modelos entre varios marcos y herramientas de ML (ONNX | Home, 2025).

SQL Server: es un sistema de gestión de base de datos relacional, desarrollado por la empresa Microsoft. El lenguaje de desarrollo utilizado es Transact-SQL (TSQL) (Microsoft SQL Server, 2025).

Tesseract: es un motor de reconocimiento óptico de caracteres para sistemas operativos. Es software libre (Tesseract (Software) - Wikipedia, La Enciclopedia Libre, 2025).
Parquet: es una tecnología de Python que consiste en un formato de almacenamiento de datos que se usa para bases de datos orientadas a columnas. Es un formato de código abierto usado para análisis de datos (Apache Parquet - Wikipedia, 2025).
Microsoft Visual Studio: es un entorno de desarrollo integrado (IDE) para Windows y macOS. Compatible con múltiples lenguajes de programación y con diferentes tipos de entornos (Microsoft Visual Studio, 2025).
TensorFlow (TF): biblioteca de código abierto de cálculo numérico, y aprendizaje automático a gran escala, Deep Learning (TensorFlow, 2025). (Crea Un Clasificador de Perros y Gatos Con IA, Python y Tensorflow - Proyecto Completo - YouTube, 2025).

2.4 Herramientas de Hardware

El uso de técnicas de aprendizaje automático como el aprendizaje profundo, y específicamente las RNC conllevan un alto costo computacional. Esto se debe a que estas redes están compuestas por un gran número de capas y neuronas, y trabajan con imágenes de alta resolución. El CPU de una computadora es capaz de realizar operaciones matemáticas a una velocidad considerable, pero las realiza de manera secuencial, una operación por núcleo, o al menos, una operación a la vez. En contraste, la Unidad de Procesamiento Gráfico (GPU), tiene muchos más núcleos, lo que les permite realizar un mayor número de operaciones en el mismo período de tiempo. La GPU, es un coprocesador diseñado específicamente para el procesamiento de imágenes. Esencialmente, al ser un procesador adicional, su función es aliviar la carga del CPU y aumentar su rendimiento. Toda esta potencia de cálculo aritmético se aprovecha en la ejecución de algoritmos de aprendizaje profundo (Técnicas de Deep Learning Aplicadas a Un Sistema de Clasificación de Objetos Para Un Recolector de Residuos Inteligente, 2024).

En la Tabla 1 se detallan las características del hardware utilizado.

Tabla 1. Hardware.

CPU 1	Intel(R) Core i7-8550U, 1.80GHz. GPU 0, Intel(R) UHD Graphics 620. GPU 1, NVIDIA GeForce MX150. RAM 16,0 GB, 2400 MHz.
CPU 2	Intel(R) Core i5-7400, 3.00GHz. GPU Intel(R) HD Graphics 630. RAM 16.0 GB, 2133 MHz.
CPU 3	Intel(R) Core i5-7400, 3.00GHz. GPU Intel(R) HD Graphics 630. RAM 12,0 GB, 2133 MHz.
CPU 4	Intel(R) Core 2 Duo, 2.80GHz. RAM 4,0 GB DDR2, 800 MHz.

Las pruebas en tiempo real vía web se realizaron con el CPU 4.

2.5 Implementación de Modelos

Se analizaron los registros obtenidos de las bases de datos en producción y se agruparon en categorías en función de la cantidad y de la calidad de las imágenes. Se habla de la calidad de las imágenes porque el proceso de inspección según los estándares exige que las fotos por unidad dañada incluyan por lo menos tres imágenes y con determinadas condiciones, y eso resulta muchas veces en la inclusión de datos como el código de barras de la unidad para identificarla en los sistemas de consultas o bien imágenes del entorno donde se encuentra la unidad (Muestra 1). Es por esto último que para filtrar los registros de imágenes obtenidos se utiliza una herramienta de reconocimiento óptico de caracteres para eliminar automáticamente todas las fotos que incluyan textos

derivados del código de barras. Otro filtro para las imágenes se hizo sobre las unidades que tienen más de un daño puesto que cuando sucede esto las imágenes se mezclan en la base de datos. Por lo mencionado anteriormente se establece como imagen de buena calidad aquella imagen que permite ver claramente el tipo de daño, o la parte del auto en la que se produce o bien el lugar donde es imputable el daño.

Muestra 1. Imágenes con código de barras descartadas.

Finalmente utilizando Python se ejecuta el etiquetado que resulta de las consultas estructuradas que se realizan online sobre las bases de datos. Luego del proceso de filtrado de datos de unidades dañadas quedan entre un 60 % y 65 % del total de datos de buena calidad. Se crearon tres conjuntos de datos extraídos de producción y fueron almacenados en disco en formato Parquet para ejecutar localmente los modelos. El primer conjunto contiene 50545 daños clasificados en 5 grupos balanceados de 10109 cada uno: “Dented”, “Scratched”, “Chipped”, “Missing”, “Others” (Muestra 2).

Muestra 2. Tipos de daños.

El conjunto 2 consta de 42270 partes dañadas dividida en 5 categorías balanceadas de 8454 ejemplares para cada grupo: “‘Front bumper”, “Rear bumper”, “Front door”, “Rear door”, “Miscellaneous” (Muestra 3).

Muestra 3. Partes de autos con daño.

Y el tercer sistema de datos extraídos fue generado con 52644 registros de lugares donde los daños fueron producidos y se categorizaron en 4 grupos balanceados de 13161 imágenes cada uno: “PSD” (Damage reported at preloading survey report), “SSTD” (Damage at some stage of transit), “STOW” (In stow damage), “NTD” (Non transit damages) (Muestra 4).

Muestra 4. Lugares donde los daños son producidos.

Cabe destacar que para los tres conjuntos de datos analizados y preprocesados se han tomado las muestras más representativas para generar las categorizaciones propuestas considerando cantidad y calidad de fotos. Usualmente las estructuras de datos de tipos de daños y partes dañadas constan de varias decenas de clasificaciones, pero no se cuenta con cantidades de imágenes suficientes para realizar el entrenamiento sobre todas las categorías. Los lugares donde se producen daños también constan de más categorías, pero para el presente estudio se tomaron las más representativas.

2.6 Evaluación de Rendimientos

En este trabajo se desarrollaron funciones en Python a través de las cuales se introdujeron diferentes formatos de entrada de datos, capas convolucionales y filtros. Una vez definidas las funciones se hicieron pruebas que fueron variando acorde a los parámetros establecidos y así se obtuvieron los mejores resultados que son los que se exponen en este artículo. Todos los modelos fueron entrenados desde cero, no se usaron técnicas de transfer learning con redes preentrenadas. Para el entrenamiento también se fueron procesando y extrayendo datos dinámicamente para ir refrescando la información y así aprovechar el procesamiento de los algoritmos con datos nuevos. Los modelos de mayor cantidad de horas demoraron 96 horas, mientras que los más eficientes y eficaces para los objetivos de este trabajo demoraron entre 8 y 12 horas máquina.

En los entrenamientos de los primeros modelos se observó un inicio de exactitud del 50%, motivo por el cual se siguió avanzando en la investigación puesto que los modelos con ese porcentaje de exactitud demostraban aprendizaje (no adivinaban). Luego se fueron modificando algunos parámetros como el tamaño de las imágenes, la cantidad de canales de entrada (color y escala de grises), y el número de epochs, y el porcentaje de exactitud fue subiendo hasta valores del 70%. Finalmente se hicieron ajustes sobre los hiperparámetros como las cantidades de capas ocultas, el número de neuronas de cada capa en cada nivel y la tasa de aprendizaje del optimizador para llegar al 91% de exactitud. Desde la creación de los modelos se implementaron técnicas de dropout (0.25) para mitigar el overfitting. Y además para reducir el underfitting se tomaron cantidades de datos más numerosos y se aplicaron técnicas de aumentación de datos.

Por otra parte, para evaluar el rendimiento de los modelos para cada uno de ellos se calcularon los valores de Precision, Recall y F1-Score. Se debe tener en cuenta que para los tres subproyectos se parte de la base de que el tipo de daño existe, o la parte dañada existe o bien el lugar donde sucedió corresponde a un daño tomado realmente. La Precision y el Recall se calcularon utilizando verdaderos positivos, falsos positivos y falsos negativos. Para las Accuracy se utilizaron verdaderos negativos. Mientras que la métrica F1-Score se calculó para cada modelo utilizando la Precision y el Recall. F1-Score

aportó generalización a los resultados debido a que tiene en cuenta el desbalance de las clases (Clasificación: Exactitud, Recuperación, Precisión y Métricas Relacionadas | Aprendizaje Automático | Google Para Desarrolladores, 2025).

3 Resultados

Se ejecutaron más de 50 modelos diferentes de RNC. Todos fueron entrenados desde cero. El entrenamiento duró aproximadamente tres meses, con tiempo promedio de 10 horas para los modelos más simples y 96 horas para los más complejos utilizando el hardware mencionado. Para reducir problemas de overfitting y de underfitting se abordó la problemática contemplando las soluciones desde la programación; y el filtrado y preprocesamiento de las bases de datos, y las correspondientes categorizaciones. Para mejorar los resultados obtenidos desde el inicio se probaron variaciones en los hiperparámetros como la cantidad de capas ocultas, el número de neuronas por capas y la tasa de aprendizaje; y esos cambios fueron modificando las métricas más relevantes como muestran los cuadros 1, 2 y 3.

Cuadro1. Métricas de modelos de clasificación de tipos de daños.

Modelo	Accuracy	Precision	Recall	F1 Score	Lr	Batch	Layer	Epoch
Tipo daños 1	64,00	0.75	0.84	0.80	0.001	32	7	32
Tipo daños 2	81,00	0.93	0.86	0.89	0.001	32	5	64
Tipo daños 3	80,00	0.93	0.87	0.90	0.001	32	5	64
Tipo daños 4	**85,00**	**0.94**	**0.93**	**0.94**	**0.001**	**64**	**5**	**64**
Tipo daños 5	51,00	0.71	0.94	0.81	0.001	256	5	64

Cuadro 2. Métricas de modelos de clasificación de partes dañadas.

Modelo	Accuracy	Precision	Recall	F1 Score	Lr	Batch	Layer	Epoch
Partes dañadas 1	81,00	0.95	0.90	0.92	0.001	32	7	32
Partes dañadas 2	90,00	0.96	0.95	0.96	0.001	32	5	64
Partes dañadas 3	91,00	0.96	0.96	0.96	0.001	32	5	64
Partes dañadas 4	**91,00**	**0.96**	**0.99**	**0.97**	**0.001**	**64**	**5**	**64**
Partes dañadas 5	78,00	0.93	0.85	0.89	0.001	256	5	64

Cuadro 3. Métricas de modelos de lugares donde se producen los daños.

Modelo	Accuracy	Precision	Recall	F1 Score	Lr	Batch	Layer	Epoch
Lugares 1	55,00	0.73	0.63	0.68	0.001	32	7	32
Lugares 2	68,00	0.77	0.84	0.80	0.001	32	5	64
Lugares 3	68,00	0.81	0.77	0.79	0.001	32	5	64
Lugares 4	**70,00**	**0.84**	**0.76**	**0.80**	**0.001**	**64**	**5**	**64**
Lugares 5	66,00	0.74	0.90	0.81	0.001	256	5	64

En los tres subproyectos se eligieron los modelos número 4 (remarcados) con capas iniciales de entrada de 16 neuronas con un máximo de 512. Esas variaciones permitieron una mejora considerable en el porcentaje de aciertos de los modelos propuestos. Los Gráficos 1, 2 y 3 detallan cada Accuracy como métrica relevante en este artículo.

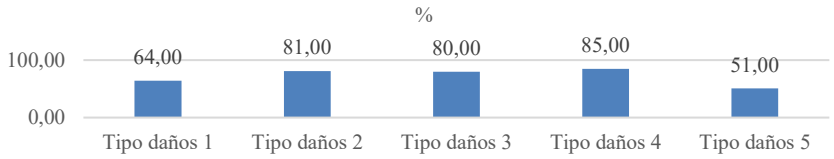

Gráfico 1. Accuracy de modelos de clasificación de tipos de daños.

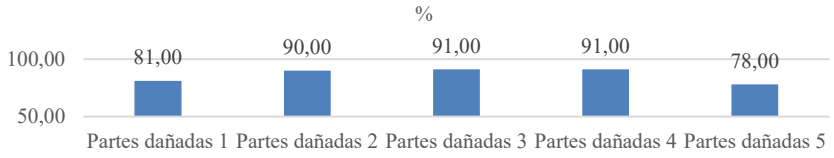

Gráfico 2. Accuracy de modelos de clasificación de partes dañadas.

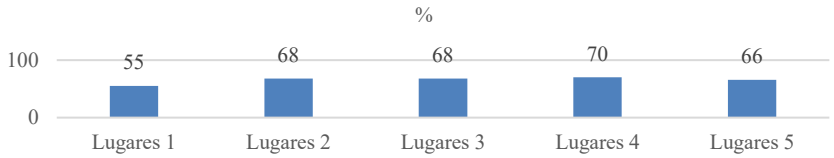

Gráfico 3. Accuracy de modelos de lugares donde probablemente se producen los daños.

Los resultados corresponden al conjunto de datos de prueba. Y en cada modelo los datos han sido particionados explícitamente para entrenamiento, evaluación y prueba.

A continuación, se presentan detalladamente los resultados obtenidos para los mejores modelos de los tres tipos de sistemas propuestos.

3.1 Modelo de clasificación de tipos de daños

Aquí se obtuvo una exactitud global del 85%. La exactitud “Missing” fue del 98%, “Dented” 95%, “Chipped” 86% y “Others” 95%; esos valores satisfacen la propuesta del proyecto. En los daños “Scratched” (70%) se evidencia la necesidad de continuar estudiando para mejorar el porcentaje, quizás con transfer learning o ciencia de datos para obtener mejores fotos de las bases de datos (Gráfico 4).

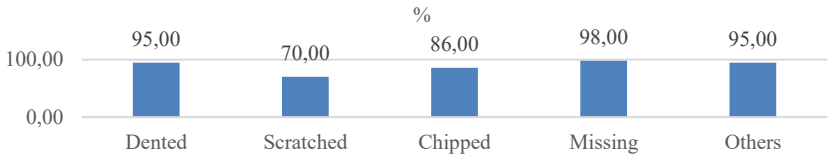

Gráfico 4. Exactitud de clasificación de tipos de daños.

En referencia a Precision (0.94), Recall (0.93) y F1-Score (0.94), se observan valores óptimos. El desempeño del algoritmo se ve en la matriz de confusión (Gráfico 5). Si bien el modelo reconoce cada clase, se puede observar que el bajo porcentaje de aciertos de “scratched” se debe a que se confunde con “chipped” y en este caso es lógico puesto que en realidad son daños muy parecidos físicamente.

Gráfico 5. Matriz de confusión de tipos de daños.

3.2 Modelo de clasificación de partes con daño

Este modelo tiene los mejores porcentajes de exactitud. Si bien se llega a una exactitud del 91%. En el caso de los "Rear bumper" (95%), "Front door" (97%), "Rear door" (97%) y "Front bumper" (93%) se cumplieron las expectativas. Y en cuanto a "Miscellaneous" (83%) es necesario estudiar para mejorar (Gráfico 6).

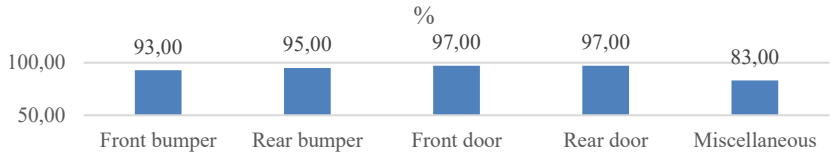

Gráfico 6. Exactitud de clasificación de partes dañadas.

En lo que respecta a Precision (0.96), Recall (0.99) y F1-Score (0.97) se observa que los valores han mejorado con respecto al modelo anterior. Se expone el desempeño del algoritmo en su matriz de confusión (Gráfico 7). Se puede observar que el porcentaje de errores en la clasificación es más bajo y equilibrado para todas las clases en general.

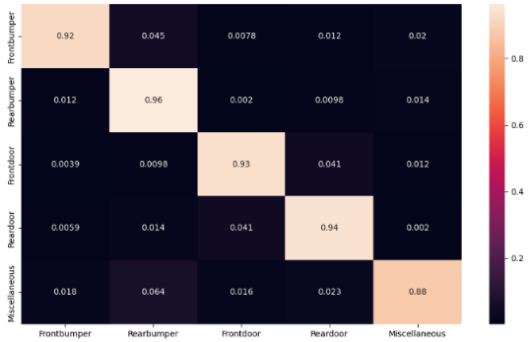

Gráfico 7. Matriz de confusión de partes dañadas.

3.3 Modelo de clasificación de lugares donde se producen daños

La clasificación de lugares es la más compleja de resolver y arroja valores bajos de exactitud global, 70% (STOW 92%, PSD 76%, SSTD 61% y NTD 60%; Gráfico 8).

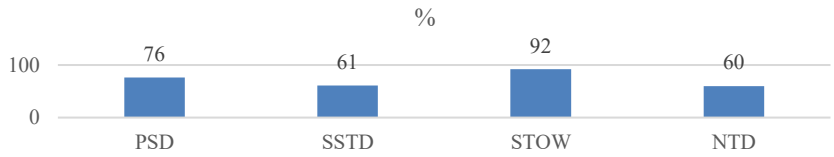

Gráfico 8. Exactitud de clasificación de lugares donde se producen los daños.

Con este modelo se intenta solucionar un problema complejo, puesto que los lugares donde se producen daños dependen del criterio de cada compañía o del personal que inspecciona, e inclusive en muchas situaciones la aceptación del daño es difícil porque no hay lineamentos claros. Las métricas de precisión (0.84) y recupero (0.77) muestran desbalanceo comparando con los otros modelos. El rendimiento del modelo con una media armónica de 0.80 indica que es un resultado regular, pero si se tiene en cuenta que uno de los objetivos de este artículo es demostrar la viabilidad técnica de soluciones con RNC, entonces el resultado se podría considerar bueno. El Gráfico 9 muestra el desempeño en la matriz de confusión. En esta matriz se ve un bajo porcentaje de aciertos en la clasificación de “SSTD” y “NTD”. En el primer caso son confundidos con los “PSD” y los “NTD”. Y en el segundo caso la confusión se presenta con los “SSTD”.

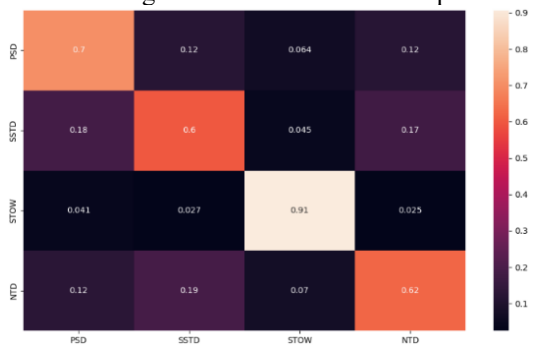

Gráfico 9. Matriz de confusión de lugares donde se producen los daños.

4 Conclusiones

El primer problema a resolver estuvo en la elección entre TF o PT. TF presenta un esquema de programación simbólico con gráficas estáticas motivo por el cual los modelos son eficientes y escalables, pero no son tan simples de programar, no son tan amigables con Python. En TF se crean los modelos, se asignan recursos y luego recién se corren los modelos con los datos disponibles. En cambio, PT ofrece un esquema de gráficos dinámicos para programar, similar a Python o sea que para proyectos con incertidumbre hacia donde hay que llevar las soluciones con RNC es más amigable. En PT resulta más simple hacer experimentos y correr rápidamente la programación, es por ello qué ha sido instalada satisfactoriamente en la comunidad científica. Sin embargo, dado que las dos bibliotecas tienen un apoyo comunitario muy fuerte, durante la implementación de los modelos se realizaron pruebas con ambas (¡PyTorch vs TensorFlow! ¿Cuál Es Mejor? - YouTube, 2025).

Otro punto clave, fue la elección entre la creación de un modelo desde cero o la creación de un modelo a partir de uno preentrenado. No se tienen limitantes ni de tiempos ni de recursos debido a que es un proyecto nuevo en la industria, es por ello que se creó este artículo con entrenamientos desde cero para cada modelo.

Con relación a la elección de los hiperparámetros para poder controlar el proceso de entrenamiento de los modelos, fue difícil encontrar los valores óptimos que se ajusten mejor y en todos los casos las mejoras, especialmente en la exactitud global de cada modelo, fueron de porcentajes de avance muy pequeños.

En referencia a los resultados obtenidos, este estudio alcanzó con éxito los objetivos propuestos. Los modelos exhibieron métricas prometedoras, con tasas de exactitud superiores al 95% y se realizaron diversas pruebas obteniéndose buenos resultados. Se atribuye este logro al meticuloso esfuerzo invertido en el análisis de datos y su estructura y en la elección de los hiperparámetros. Se construyó un dataset diverso, que abarcó una variedad que contempla las situaciones más representativas donde se encuentran los daños en vehículos, y además se pudo contar con una cantidad significativa de datos y fotos cuidadosamente seleccionadas. Actualmente se está implementado un proyecto donde se ve en línea si la predicción de datos coincide con la clasificación real vía web (Austral Marine Services, 2025). Si bien los tipos de daños de mejor porcentaje de clasificación son del 85% de exactitud total, se evidencia la dificultad para identificar los “Scratched”. En cambio, en las partes de vehículos dañados los porcentajes mejoran considerablemente llegando a un 91% de exactitud global, pero también se debe mejorar en “Miscelleanous”. Y con respecto a la identificación de los lugares donde los daños fueron producidos, este estudio muestra que se debe continuar investigando y buscando una forma de mejorar los porcentajes de exactitud global (70%). Posiblemente para esto último una solución podría ser a través de tecnología de transfer learning.

Con relación a tareas a futuro se pretende mejorar los resultados obtenidos e integrar los modelos con sistemas robóticos para detección y clasificación de imágenes. En el presente la búsqueda de daños en lugares bajos, como por ejemplo en los paragolpes se hace manualmente con herramientas como espejos. También se considera abordar la implementación de un sistema de inspección con celulares que asistan online a los peritos para detectar y clasificar los daños conectados directamente a las bases de datos centrales publicando los resultados de las inspecciones en tiempo real.

Finalmente, observando la industria en general se podría proponer la normalización (especialmente para toma de fotos) de los sistemas de inspección para que el control total de los autos que salen de producción y llegan a su destino final sea más eficiente.

5 Referencias

Anaconda - Advance AI with Open Source | Anaconda. (2025). https://www.anaconda.com

Apache Parquet - Wikipedia. (2025). https://en.wikipedia.org/wiki/Apache_Parquet

Austral Marine Services. (2025). http://www.australmarine.com.ar/Index_en.html

Clasificación: Exactitud, recuperación, precisión y métricas relacionadas | Aprendizaje automático | Google para desarrolladores. (2025). https://developers-google-

com.translate.goog/machine-learning/crash-course/classification/accuracy-precision-recall?_x_tr_sl=en&_x_tr_tl=es&_x_tr_hl=es&_x_tr_pto=tc

Google Colab. (2025). https://colab.google/

Comprender las redes neuronales convolucionales (CNN). (2025). https://es.innovatiana.com/post/convolutional-neural-network

Crea un clasificador de perros y gatos con IA, Python y Tensorflow - Proyecto completo - YouTube. (2025). https://www.youtube.com/watch?v=DbwKbsCWPSg&t=1817s

Flores, H. D. & Neil, C. G. (2023). Detección de daños con visión artificial en inspecciones marítimas: un mapeo sistemático de la literatura. https://sedici.unlp.edu.ar/handle/10915/164809

Flores, H., Garcia-Martinez, R., Fernandez, E., Merlino, H., Rodrguez, D., & Britos, P. (2022). Detección de Patrones para la Prevención de Daños y/o Averías en la Industria Automotriz. http://sedici.unlp.edu.ar/handle/10915/21204

Flórez, C., Sandoval, O., Agronómica, D. H.-A. (2022). Procesamiento de imágenes para reconocimiento de daños causados por plagas en el cultivo de Begonia semperflorens (flor de azú-car). https://revistas.unal.edu.co/index.php/acta_agronomica/article/view/42657

Larcher, L. I., Juárez, P. M., Ruggeri, A. I., Biasoni, E. M., Cattaneo, C. A., Villalba, G. A., García Garino, C. G., Mirasso, A. E., Storti, M. A., Tornello, M. E., & Mendoza, A. (2013). Ponderación de calidad en frutas usando técnicas de visión artificial para la estimación de daños. https://amcaonline.org.ar/ojs/index.php/mc/article/view/4498

Microsoft SQL Server. (2025). https://es.wikipedia.org/wiki/Microsoft_SQL_Server

Microsoft Visual Studio. (2025). https://es.wikipedia.org/wiki/Microsoft_Visual_Studio

ONNX | Home. (2025). https://onnx.ai/

Practical Deep Learning for Coders - YouTube. (2025). https://www.youtube.com/watch?v=8SF_h3xF3cE&list=PLfYUBJiXbdtSvpQjSnJJ_PmDQB_VyT5iU&index=2

Project Jupyter | Home. (2025). https://jupyter.org/

PyTorch. (2025). https://pytorch.org/

¡PyTorch vs TensorFlow! ¿Cuál es mejor? - YouTube. (2025). https://www.youtube.com/watch?v=YRA6ArsGlcw&t=19s

Russo, C., Ramón, H., Serafino, S., Cicerchia, B., Sarobe, M., Balmer, A., Álvarez, E., Luengo, P., Useglio, G., & Faroppa, M. (2022). Visión artificial aplicada en agricultura de precisión. http://sedici.unlp.edu.ar/handle/10915/68308

Salina, M., Pezet, B., Osés, L., Cappelletti, M. Á., Osio, J. R., & Morales, M. (2024). Técnicas de Deep Learning aplicadas a un sistema de clasificación de objetos para un recolector de residuos inteligente. http://sedici.unlp.edu.ar/handle/10915/168756

Santome Jiménez, B. J. (2020). Impacto de los riesgos y sobrecostos logísticos presentados en el despacho aduanero de los vehículos nuevos importados, ingresados por la aduana marítima del Callao 2013 al 2017.

Técnicas de Deep Learning aplicadas a un sistema de clasificación de objetos para un recolector de residuos inteligente. (2025). https://sedici.unlp.edu.ar/handle/10915/168756

TensorFlow. (2025). https://www.tensorflow.org/?hl=es-419

Tesseract (software) - Wikipedia. (2025). https://es.wikipedia.org/wiki/Tesseract_(software)

Uso de PyTorch para entrenar el modelo de clasificación de imágenes | Microsoft Learn. (2025). https://learn.microsoft.com/es-es/windows/ai/windows-ml/tutorials/pytorch-train-model